UNIVERSIDAD AUTONOMA DE MADRID

ESCUELA POLITECNICA SUPERIOR

Grado en Ingeniería Informática

TRABAJO FIN DE GRADO

Identificación de líneas de carreteras y cálculo de trayectorias mediante análisis de imágenes

Álvaro Morón Elorza
Tutor: Manuel Sánchez-Montañés

Julio 2020

Identificación de líneas de carretera y cálculo de trayectoria mediante análisis de imágenes

AUTOR: Álvaro Morón Elorza
TUTOR: Manuel Sánchez-Montañés

Dpto. Ingeniería Informática
Escuela Politécnica Superior
Universidad Autónoma de Madrid
Julio de 2020

Resumen (castellano)

Este Trabajo de Fin de Grado tiene como objetivo diseñar un sistema para la identificación de líneas de carretera y el cálculo de la trayectoria de un vehículo mediante algoritmos de análisis y tratamiento de imágenes aplicados a un algoritmo.

El sistema utilizará imágenes preprocesadas, procedentes de una cámara situada en una posición centrada del salpicadero y techo de un vehículo, en el sentido de la marcha. Se han diseñado normalizaciones y manipulaciones a los canales RGB y HSL, transformaciones de perspectiva y algoritmos de estimación de posición de líneas, y mediante la posterior aplicación del algoritmo Sliding Window Technique, se detectarán las marcas de trazado de la carretera para un posterior análisis de las ecuaciones, denotadas por estas líneas, y proceder con el cálculo de la trayectoria deseada del vehículo. Se ha creado un evaluador que calculará el porcentaje de trayectoria coincidente entre los fotogramas contiguos para la validación de la trayectoria calculada, finalizando la ejecución del algoritmo con la obtención de la trayectoria más adecuada para cada fotograma. El conjunto de datos utilizado para las pruebas del sistema es único, recopilado para la realización de este TFG.

Como conclusión del trabajo, se aporta una evolución de las diferentes pruebas que se han realizado a lo largo del desarrollo del sistema, se discuten los resultados obtenidos del algoritmo, así como diferentes mecanismos de mejora implementados y se marcan posibles aproximaciones futuras para la mejora del sistema.

Palabras clave

Calibración de cámara, RGB, HSL, umbralización Otsu, umbralización Sobel, región de interés, transformación de perspectiva, algoritmo Sliding Window Technique.

Abstract (English)

This Bachelor Thesis aims to design a system for the identification of road lines and the calculation of the trajectory of a vehicle through the analysis and treatment of images applied to an algorithm.

The system will use preprocessed images taken from a camara located in a centered position on the dashboard and the roof of a vehicle, in the direction of travel. Image normalizations, manipulations to the RGB and HSL channel and perspective transformations have been designed, and through the subsequent application of the Sliding Window Technique algorithm, road marking will be detected for a later analysis of the equations, denoted by these lines, and proceed with the calculation of the desired trajectory of the vehicle. An evaluator has been created that will calculate the percentage of coincident trajectory between contiguous frames for the validation of the calculated trajectory, finishing the algorithm by obtaining the most adequate trajectory for each frame. The dataset used is unique, as it has been created for the realization of this Bachelor Thesis.

As a conclusion of this work, an evolution of the different tests that have been carried out through the development of the system is provided, the results obtained from the algorithm are discussed, as well as different improvements implemented and possible future approaches for the improvement of the system.

Keywords

Camera calibration, RGB, HSL, Otsu segmentation, Sobel segmentation, region of interest, perspective transformation, Sliding Window Technique algorithm

Agradecimientos

Agradecer a mi tutor Manuel la oportunidad de poder realizar este Trabajo de Fin de Grado con él. Gracias a mis amigos por compartir conmigo la experiencia única que es la universidad.

A mi familia, por todo el apoyo que me ha dado durante estos últimos cuatro años ayudándome a mantener el rumbo correcto, y a María, sin la cual no hubiese llegado tan lejos.

INDICE DE CONTENIDOS

INDICE DE FIGURAS

1 Introducción

1.1 Motivación

Desde sus orígenes, el ser humano siempre ha buscado la manera de automatizar sus procesos y tareas, facilitándose la vida, ahorrando tiempo y esfuerzo y reduciendo costes de producción.

Con la aparición de los microprocesadores la automatización de tareas experimentó una revolución, abriendo las puertas a trabajos que antes eran considerados como imposibles de realizar por una máquina, como pueden ser la ayuda de pilotaje de un transbordador espacial o la creación de robots humanoides completamente funcionales. Las posibilidades de nuevas automatizaciones siguen aumentado año tras año debido a las mejoras y avances que se consiguen en microprocesadores y en el tratamiento de los materiales de los cuales están compuestos, así como por la reducción de sus costes y el desarrollo de tecnologías que lo hacen posible.

En los últimos años, los recursos destinados a la investigación de la conducción autónoma han aumentado exponencialmente. De manera similar a la carrera espacial de los años 60 entre las grandes potencias, existe una competición entre las principales marcas de automoción por capitalizar el liderazgo en el desarrollo de nuevos algoritmos y tecnologías que permitan una conducción autónoma más segura y confiable.

A pesar de los continuos avances en los sistemas de seguridad de los vehículos, más de 1,3 millones de personas fallecen anualmente en el mundo por accidentes de tráfico [16], muchos debidos a errores humanos por distracción, temeridad o cansancio, la mayoría en países poco desarrollados. La popularización de sistemas de conducción autónoma fiables y asequibles ayudaría a mitigar en parte esa altísima tasa de siniestralidad, equivalente a un muerto cada 25 segundos. [17]

Pero además hay también un doble componente vocacional para la realización de este trabajo: Desde una temprana edad me he sentido fascinado simultáneamente por la automoción y por el mundo de la informática, haciendo de esta última mi carrera profesional. Por lo tanto, hacer un Trabajo de Fin de Grado tratando de acercar dos de las pasiones de mi vida en un trabajo de investigación ha supuesto una motivación extra.

1.2 Objetivos

Este Trabajo de Fin de Grado consiste en un caso de uso orientado a la conducción autónoma, como es la determinación de la trayectoria de un vehículo mediante el reconocimiento de imágenes y la adaptación de un algoritmo, utilizando librerías de código abierto. La finalidad perseguida es la detección de las marcas de trazado de una carretera mediante procesamiento y manipulación de imágenes, determinando como resultado la trayectoria adecuada a ese trazado para un vehículo que circule por una vía.

Este sistema está basado en el desarrollo de [14], implementando en cada una de las etapas mejoras en cuanto a la ejecución del algoritmo, añadiendo técnicas de preprocesamiento de imágenes, y la creación de un posterior evaluador para la validación de los resultados que produce. Una de las metas de la implementación de este algoritmo y sus mejoras es tratar de calcular la trayectoria del vehículo con una frecuencia de 30 fotogramas por segundo, de

manera estable a lo largo del tiempo de procesado para que sea compatible y coordinado con las frecuencias de grabación de fotogramas de las cámaras.

Como finalización del trabajo se presentan las conclusiones sobre el estudio realizado, y se proponen campos de investigación alternativos para una posible evolución y mejora del sistema. También se añade un extracto y comparativa respecto a otro trabajo de investigación en curso, con objetivos similares.

1.3 Organización de la memoria

La memoria consta de los siguientes capítulos, presentando algunos de éstos subsecciones para explicar de una forma más detallada algunos aspectos del capítulo:

- Capítulo 1: Introducción.
- Capítulo 2: Estado del arte.
- Capítulo 3: Diseño y desarrollo.
- Capítulo 4: Pruebas y resultados.
- Capítulo 5: Conclusiones y trabajo futuro.

2 Estado del arte

2.1 Introducción

Las innovaciones que han surgido durante los últimos años en técnicas de tratamiento de imágenes, así como el aumento de la capacidad de procesamiento de los microprocesadores a la hora de trabajar sobre éstas y el desarrollo de librerías de código abierto diseñadas específicamente para la manipulación de imágenes, perfeccionan y nos acercan cada día más al desarrollo de sistemas de conducción autónomos.

Actualmente, hay dos estrategias relacionadas con la identificación de las líneas de carretera y el cálculo de las trayectorias mediante algoritmos de análisis y tratamiento de imágenes y la manipulación de éstas: Una primera está basada en la creación de redes neuronales para realizar este trabajo [1][2]. La segunda, donde algoritmos son específicamente diseñados para estos propósitos sin el uso de redes neuronales. [3][4]

Ambas corrientes presentan una dificultad muy elevada a la hora de desarrollar algoritmos robustos, dada la diversidad del entorno y la variabilidad de sus condiciones, entre las cuales cabría destacar iluminación, visibilidad, condiciones atmosféricas e imperfecciones en el trazado y en las marcas viales, que podrían afectar a la toma de imágenes y su reconocimiento posterior.

Por el momento, la mayor parte de los esfuerzos se están invirtiendo en investigación con redes neuronales y deep learning. En [23], B. T. Nugraha, S. Su y Fahmizal demuestran cómo mediante la aplicación de redes neuronales se puede detectar de manera precisa las líneas de una carretera, los vehículos circundantes al que se está estudiando, y una trayectoria deseada adecuada en autovías. Un ejemplo claro de los esfuerzos se está realizando en este campo con redes neuronales es [22], donde Yuchi Tian, Kexin Pei, Suman Jana, y Baishakhi Ray desarrollaron un algoritmo para detectar comportamientos erróneos de vehículos conducidos por redes neuronales que podrían producir accidentes, mediante la prueba de casos extremos en variables del entorno.

En [26], Z. Chen y X. Huang, omiten los pasos de descomposición del problema en objetivos, como detección de carriles, calcular trayectorias o controlar la dirección del vehículo mediante la aplicación de redes neuronales convolucionales, concluyendo que el modelo de extremo a extremo predice correctamente la dirección del vehículo, así como el ángulo de giro del volante después de entrenar las redes. En el artículo, se comenta brevemente el problema del conjunto de datos de validación de la dirección del volante, puesto que, al ser los datos provenientes de una entrada humana, no son totalmente precisos, marcando una vez más la posibilidad de errores humanos en la conducción.

Otras aproximaciones con redes neuronales más avanzadas como [27] permiten simular las respuestas cerebrales, visual y motora, mediante la aplicación de redes neuronales recurrentes y un conjunto de validación etiquetado. Los autores concluyen que, mediante el modelo desarrollado, el sistema es capaz de aprender de la conducción humana hasta el punto de imitarla con movimientos como cambios de carriles, seguimiento de vehículos, incluso alcanzando la conducción libre.

En este trabajo se desarrollará un algoritmo de detección de líneas de carretera que mediante el tratamiento y procesado de las imágenes obtenga la ecuación del trazado de la carretera, caracterizando así la trayectoria a seguir por el vehículo. Las técnicas y transformaciones más relevantes que se han encontrado en la literatura relacionadas con esto son:"

2.2 Transformación de perspectiva

En [5], George Wolberg realiza un estudio con una amplia variedad de técnicas para la transformación de imágenes digitales, en el cual menciona [7], donde Paul S. Heckbert redacta como descubrió la transformación de perspectiva en el año 1986. En su estudio, Heckbert redacta de manera extensa cómo realizar una transformación de perspectiva sobre imágenes. Esta transformación está diseñada para la conversión de una imagen en tres dimensiones de un único canal, por ejemplo, en una escala de grises, a una imagen en dos dimensiones sobre la que se puedan detectar regiones u objetos de una manera más sencilla que en una imagen en tres dimensiones.

2.2.1 Calibración de cámara

Para la calibración de una cámara se utiliza frecuentemente la calibración mediante una imagen de un tablero de ajedrez como en el trabajo de Alessandro Bevilacqua, Alessandro Gherardi y Ludovico Carozza [6]. En el trabajo mencionado se reconstruye el tablero y sus características mediante una transformación de perspectiva, y se obtiene una matriz de homografía para el cálculo de la transformación sobre la imagen original, pudiendo corregir la calibración de la cámara.

Esta calibración sirve también para corregir diferentes efectos indeseados generados por una cámara al tomar una fotografía. En este estudio se trabajará con la calibración de la cámara para corregir un efecto conocido como distorsión radial positiva.

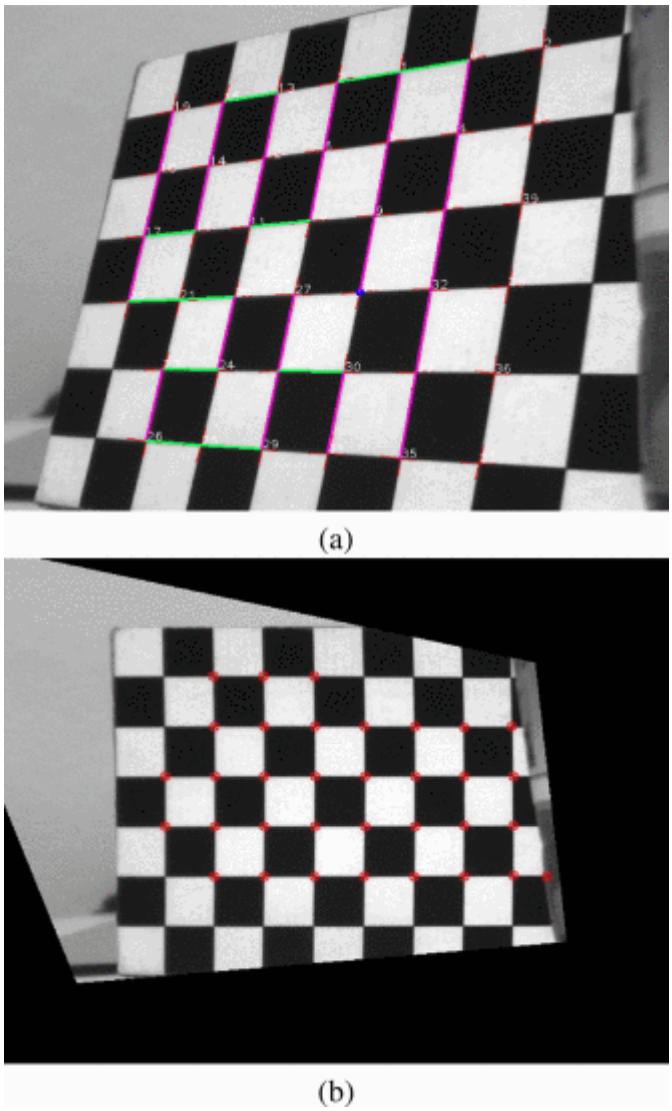


Figura 2-1: Figura extraída de [6] para la calibración de una cámara con tablero de ajedrez mediante una transformación de perspectiva

2.3 Normalización de imágenes y espacios de colores

Una gran problemática presente en el uso de algoritmos de tratamiento de imágenes es la gran variabilidad del entorno en el que se pueden tomar las fotografías. Por tanto, se tiende a normalizar los diferentes canales de color de una imagen para minimizar la posible alteración de los resultados. Actualmente, hay una gran variedad de canales de colores y, como se puede observar en [8], las regiones de interés de los algoritmos de detección de líneas tienen normalmente un preprocesado sobre los canales RGB y HSL. Puesto que en este trabajo se usarán ambos canales, se describe a continuación el estado del arte de ambos. Una de las características más destacables de las transformaciones entre espacios de colores es que se ejecutan de manera extremadamente rápida en hardware en tiempo real [11].

2.3.1 Espacio y normalización RGB

En [9], se realiza una normalización sobre el canal RGB para reducir las distorsiones causadas por las variaciones de iluminación, al mismo tiempo que se usa para obtener mejores resultados en el modelado de color. Esta técnica también es utilizada en los algoritmos de detección de líneas de carretera con redes neuronales [10] donde, al igual que

en [9], se calcula la media de cada canal normalizada entre la suma de todos sus canales, obteniendo así una imagen normalizada sobre la que trabajar.

2.3.2 Espacio HSV

En [11], se experimenta con detección de líneas de carreteras tras la transformación al espacio de color HSV. Los autores concluyen la imposibilidad de la detección de marcas del trazado de la carretera únicamente mediante la transformación del espacio de color RGB al espacio de color HSV, demostrando la posibilidad de detectar las marcas de trazado aplicando una combinación de transformaciones a HSV y normalizaciones en conjunto con otros algoritmos de manipulación de imágenes.

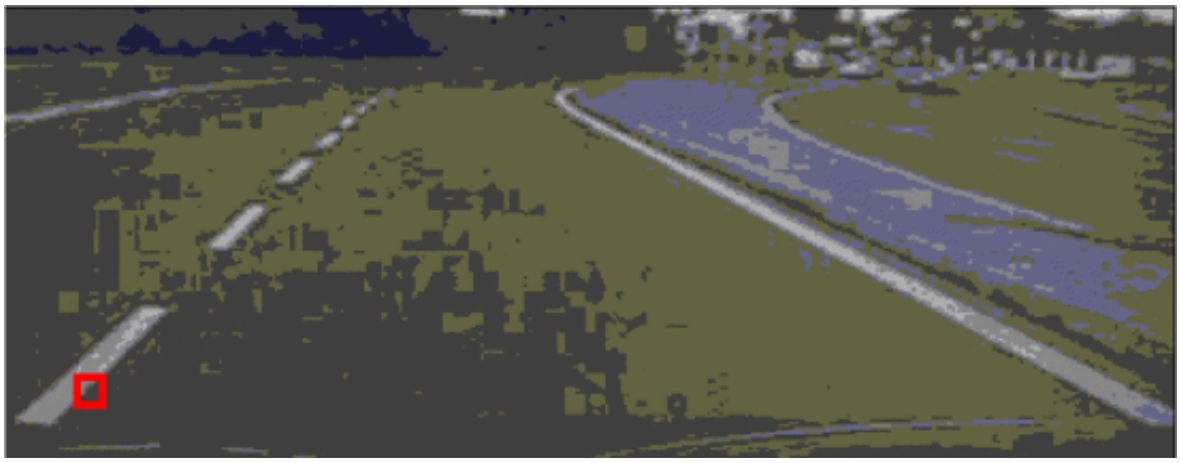


Figura 2-2: Figura extraída de [11] para la transformación al espacio de color HSV y la posibilidad de detección de marcas de trazado en este espacio

2.4 Umbralización Otsu y Sobel

Se procede a citar y explicar el estado del arte de la umbralización Otsu y la umbralización Sobel.

2.4.1 Umbralización Otsu

Xiangyang Xu, Shengzhou Xu, Lianghai Jin y Enmin Song en [25] determinan que la umbralización Otsu es igual al promedio de la media de los niveles de las dos clases divididas mediante este método de umbralización. Por ello, proponen una variación a la umbralización Otsu planteando limitaciones de rango cuando la diferencia entre el fondo de la imagen y el objeto sea significativa.

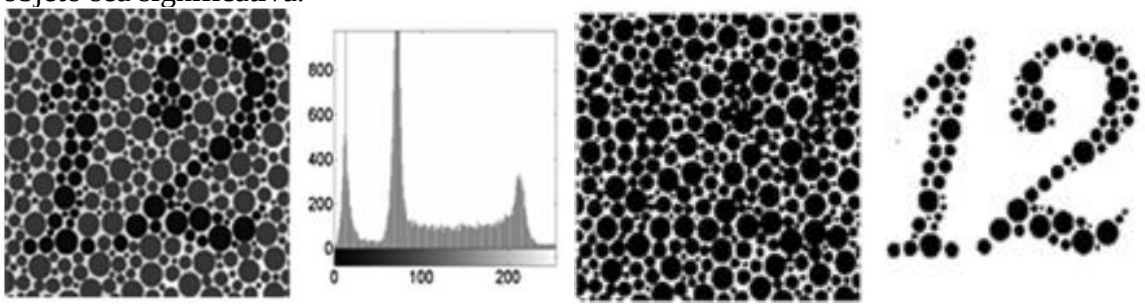


Figura 2-3: Figura extraída de [25] mostrando de izquierda a derecha: imagen original, histograma, umbralización Otsu, variación de umbralización propuesta en [25]

2.4.2 Umbralización Sobel

En [24], Wenshuo Gao, Xiaoguang Zhang, Lei Yang y Huizhong Liu, muestran una nueva aproximación de la umbralización Sobel junto con eliminación de ruido. Esta aproximación permite una mejor detección de bordes y contornos que los métodos tradicionales, como el detector de bordes Canny o el detector de bordes Laplaciano. En el estudio realizado, se puede observar cómo la aplicación de esta variación permite la eliminación de ruido de forma eficaz y produce unos resultados muy apropiados en la detección de bordes.

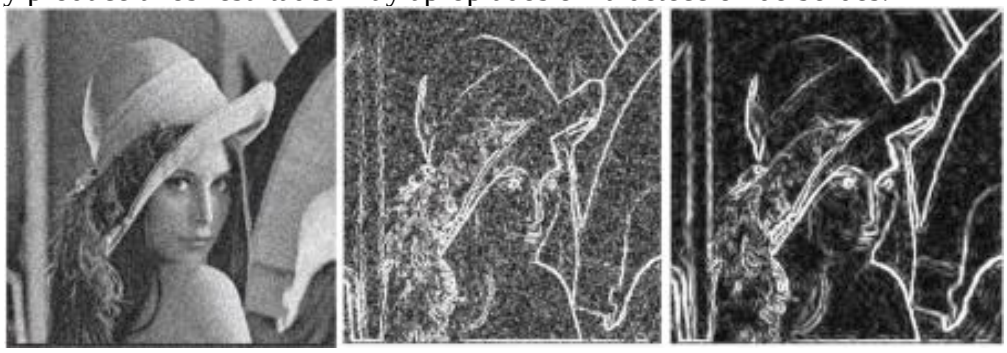


Figura 2-4: Figura extraída de [24] mostrando de izquierda a derecha: imagen original, umbralización Sobel, aproximación de umbralización Sobel propuesta en [24]

2.5 Algoritmo de ventanas deslizantes

Como se redacta en [12] Yanjun Fan, Weigong Zhang y Xu Li, investigan sobre la aplicación del algoritmo de ventanas deslizantes (Sliding Window Technique en inglés), SWT de ahora en adelante.

El algoritmo SWT se basa en la superposición de regiones cuadradas, llamadas ventanas, cuyo trabajo es buscar en una imagen dada regiones de interés o características destacadas que resulten interesantes para un propósito, en este caso, la búsqueda de líneas de trazado de carretera. Una de las ventajas del algoritmo SWT es su rapidez al aplicarlo sobre colecciones o listas, pudiendo bajar la complejidad de algoritmos de búsqueda por fuerza bruta de O(n^2) a O(n).

En [12] se concluye que la aplicación de este algoritmo a la detección de líneas de la carretera es una aproximación acertada que, aplicada junto a diferentes técnicas de tratamiento y manipulación de imágenes, produce unos resultados apropiados para la tarea de identificación de líneas carretera.

A su vez, en [13], los autores hacen uso del algoritmo SWT para la detección de las marcas de trazado de una carretera tras haber aplicado una transformación de perspectiva, produciendo unos resultados íntegros a 20 fotogramas por segundo, y aproximándose así a la barrera psicológica de los 30 fotogramas por segundo, frecuencia de grabación de fotogramas que hasta hace no mucho tiempo eran empleadas en el cine o en la televisión.

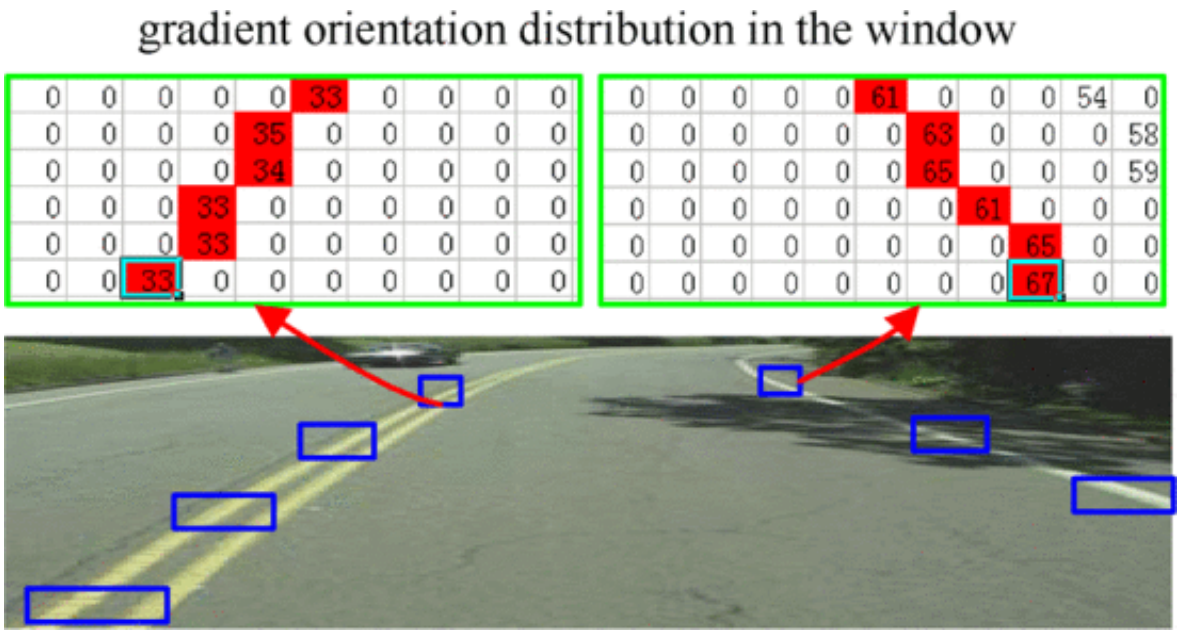


Figura 2-5: Figura extraída de [12] que ilustra la aplicación del algoritmo SWT

3 Diseño y desarrollo

Dentro de este capítulo de la memoria se va a explicar el diseño del sistema implementado y su desarrollo. Posteriormente, se dará una explicación detallada de cada una de las etapas donde se explicarán los fundamentos de los métodos y algoritmos utilizados, así como se expondrá mediante ejemplos gráficos el funcionamiento de cada una de las etapas.

El sistema se basará en un canal, o pipeline en inglés. Estará formado por diferentes etapas:

- Etapa 0: Calibración de la cámara.
- Etapa 1: Corrección de la distorsión de la imagen.
- Etapa 2: Normalización RGB y filtrado HSL.
- Etapa 3: Aplicación de umbrales Otsu y Sobel.
- Etapa 4: Identificación de la región de interés.
- Etapa 5: Transformación de perspectiva.
- Etapa 6: Detección de marcas de trazado de la carretera mediante la aplicación del algoritmo SWT.
- Etapa 7: Cálculo de la trayectoria del vehículo.

Al finalizar la séptima etapa, el sistema habrá procesado un fotograma del vídeo entrante al pipeline, y superpondrá la trayectoria calculada por éste a al fotograma original.

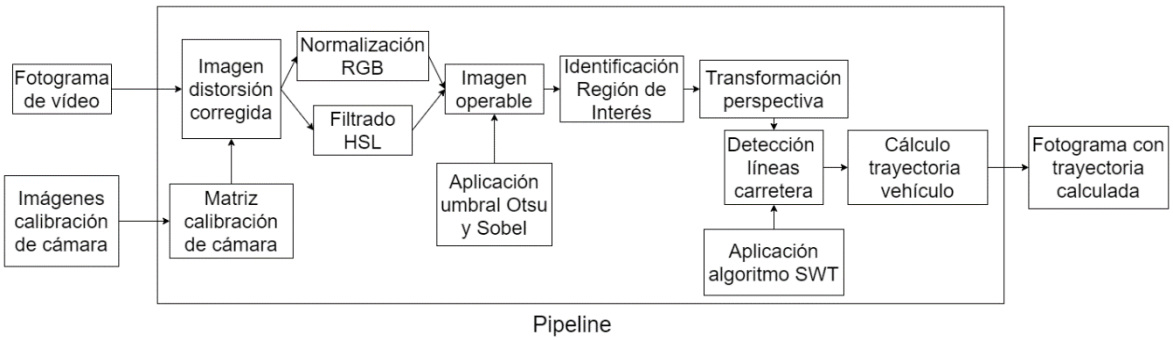


Figura 3-1: Pipeline del sistema

El procesamiento de este sistema se ha realizado en un procesador Intel Core i7-1065G7, de 4 núcleos y 8 subprocesos a una frecuencia máxima de 3,9GHz. El computador donde se han ejecutado estas pruebas también cuenta con una memoria RAM de 16GB SDRAM DDR4 a 2666MHz y una unidad de almacenamiento SSD de 1TB PCIe NVMe M.2.

Los vídeos utilizados como entrada de datos han sido tomados con una cámara GoPro Hero 7 White. Estos vídeos se han realizado a una resolución de 1920 píxeles de ancho por 1440 píxeles de alto a 30 fotogramas por segundo, aunque para agilizar su procesamiento se ha reducido la calidad de imagen posteriormente a 640 píxeles de ancho por 480 píxeles de alto.

Inicialmente, la captura de las fotografías y los vídeos necesarios como entrada del sistema se efectuó desde el salpicadero del vehículo. Tras algunas pruebas, se observó que los resultados eran mucho más consistentes al situar la cámara en el techo del vehículo.

El sistema de pipeline se ha integrado como se muestra posteriormente, donde paso a paso se avanza hacia el cálculo de la trayectoria del vehículo. El código ha sido desarrollado en el lenguaje de programación Python, en la versión 3.8.2 [20], y usando las librerías de código abierto NumPy (v. 1.18.4) [18], OpenCV (v. 4.2.0) [15], Scikit-video (v. 1.1.10) [19] y Matplotlib (v.3.2.1) [21].

3.1 Etapa 0: Calibración de la cámara

La etapa cero únicamente se ejecutará una vez, donde se calculará una matriz de homografía para corregir la distorsión radial positiva, también conocida como distorsión de barril, que ocurre en la cámara con la que este sistema ha sido implementado. Este cálculo se realizará usando un modelo de cámara estenopeica, donde se formará una vista de escena proyectando puntos en tres dimensiones sobre el plano de la imagen utilizando una transformación de perspectiva.

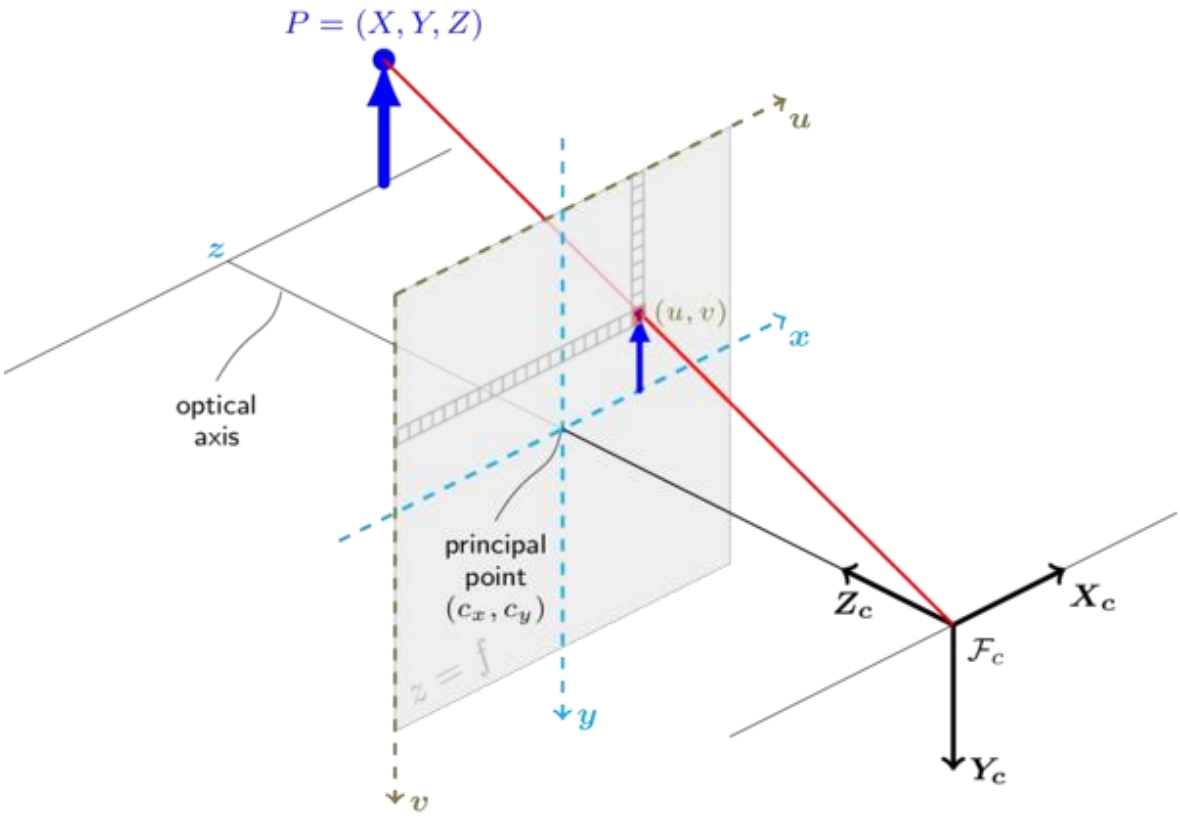


Figura 3-2: Figura extraída de [15] mostrando el modelo de cámara estenopeica

Siendo (X, Y, Z) las coordenadas del punto en tres dimensiones en el espacio de coordenadas, (u, v) las coordenadas de proyección del punto, A la matriz de la cámara, (c_x, c_y) el punto central de la imagen, y f_x y f_y las distancias focales expresadas en pixeles. Como se puede observar en [15] la matemática que demuestra la figura anterior es la siguiente:

$$s\, m' = A\, [R|t]\, M'$$

$$s\begin{pmatrix} u \\ v \\ 1 \end{pmatrix} = \begin{pmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{pmatrix}\begin{pmatrix} r_{11} & r_{21} & r_{31} & t_1 \\ r_{12} & r_{22} & r_{32} & t_2 \\ r_{13} & r_{23} & r_{33} & t_3 \end{pmatrix}\begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix}$$

En el caso de este trabajo, esta matemática es ampliada por la distorsión y el escalado en la imagen, por lo que el modelo anterior se extiende:

$$\begin{pmatrix} x \\ y \\ z \end{pmatrix} = R\begin{pmatrix} X \\ Y \\ Z \end{pmatrix} + t$$

Despejando las ecuaciones, obtenemos (u, v) que son las coordenadas de la proyección del punto (X, Y, Z):

$$u = f_x\, x'' + c_x$$
$$v = f_y\, y'' + c_y$$

Donde:

$$x' = \frac{x}{z} \qquad y' = \frac{y}{z}$$

$$x'' = x'\frac{1 + k_1\, r^2 + k_2\, r^4 + k_3\, r^6}{1 + k_4 r^2 + k_5\, r^4 + k_6\, r^6} + 2p_1 x' y' + p_2(r^2 + 2x'^2)$$
$$y'' = y'\frac{1 + k_1\, r^2 + k_2\, r^4 + k_3\, r^6}{1 + k_4 r^2 + k_5\, r^4 + k_6\, r^6} + 2p_2 x' y' + p_1(r^2 + 2y'^2)$$

Siendo $r^2 = x'^2 + y'^2$, $k_1, k_2, k_3, k_4, k_5, k_6$ los coeficientes radiales de distorsión, y p_1, p_2 los coeficientes tangenciales de distorsión.

Con las funciones declaradas en [15], obtenemos A, la matriz de la cámara, y $k_1, k_2, k_3, k_4, k_5, k_6, p_1, p_2$, los coeficientes de distorsión.

3.2 Etapa 1: Corrección de la distorsión de la imagen

Con los valores calculados en la etapa cero, se aplica la matriz de la cámara y los coeficientes de distorsión para corregirla y obteniendo una imagen sin distorsiones. Estos se han calculado con imágenes de un tablero de ajedrez encontrando los puntos interiores de los cuadrados del tablero y resolviendo las ecuaciones mencionadas en la etapa anterior.

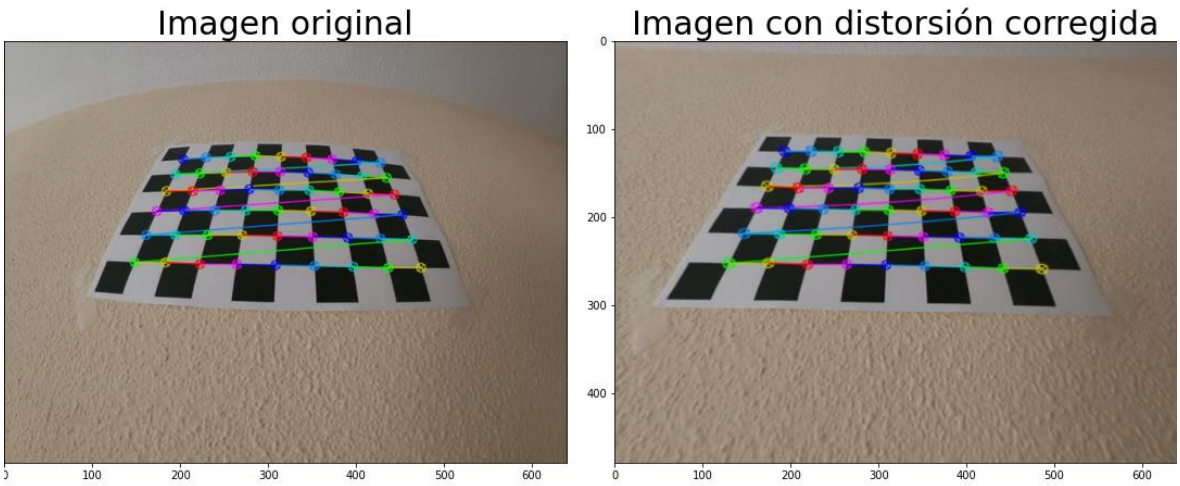


Figura 3-3: Ejemplo corrección de distorsión tablero de ajedrez

Esta misma matriz de cámara y los valores para la corrección de la distorsión permiten corregir la distorsión de la cámara en cada uno de los fotogramas del vídeo. Aquí un ejemplo de corrección en un fotograma aleatorio.

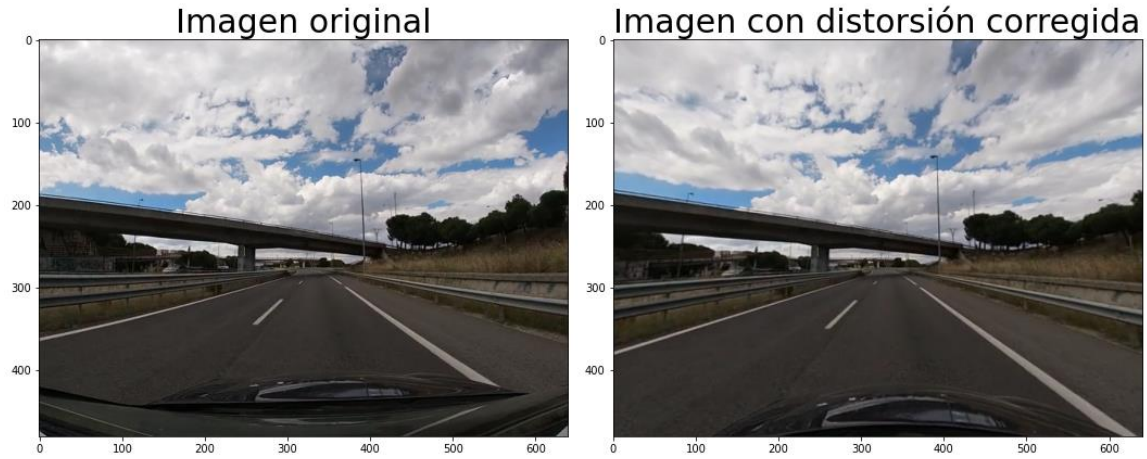


Figura 3-4: Ejemplo corrección de distorsión radial positiva sobre fotograma del vídeo

3.3 Etapa 2: Normalización RGB y filtrado HSL

En la etapa dos, se precalcula la media de cada canal RGB a lo largo de todo el vídeo para posteriormente normalizar cada fotograma con estos valores. Realizando esta normalización se logra reducir notoriamente los reflejos constantes producidos en el parabrisas y en el capó por la posición del sol, y se resaltan las líneas del trazado de la carretera.

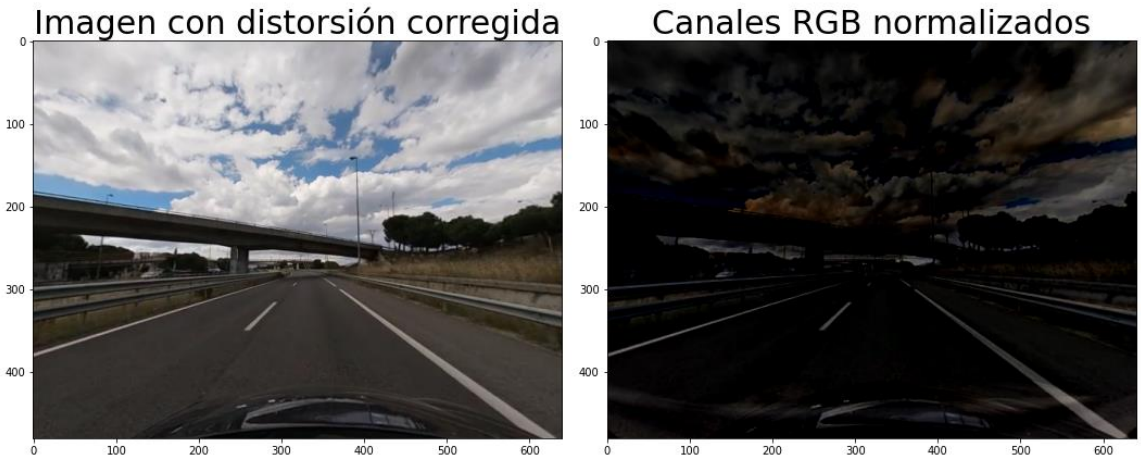


Figura 3-5: Ejemplo normalización de canales RGB

En el caso del filtrado HSL, al igual que con los canales RGB, se precalcula la media sobre el canal S para identificar la región del cielo, y con esa media, cualquier píxel que supere este valor, será considerado cielo y descartado para el tratamiento de la imagen.

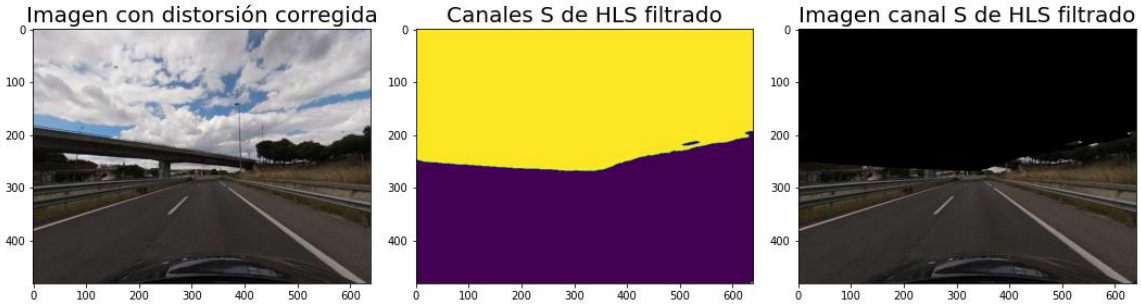


Figura 3-6: Ejemplo filtrado de canal S de espacio de colores HSL

Con la combinación de ambas, se consigue descartar gran parte del cielo, resaltar las marcas del trazado de la carretera y reducir los reflejos continuos en el parabrisas.

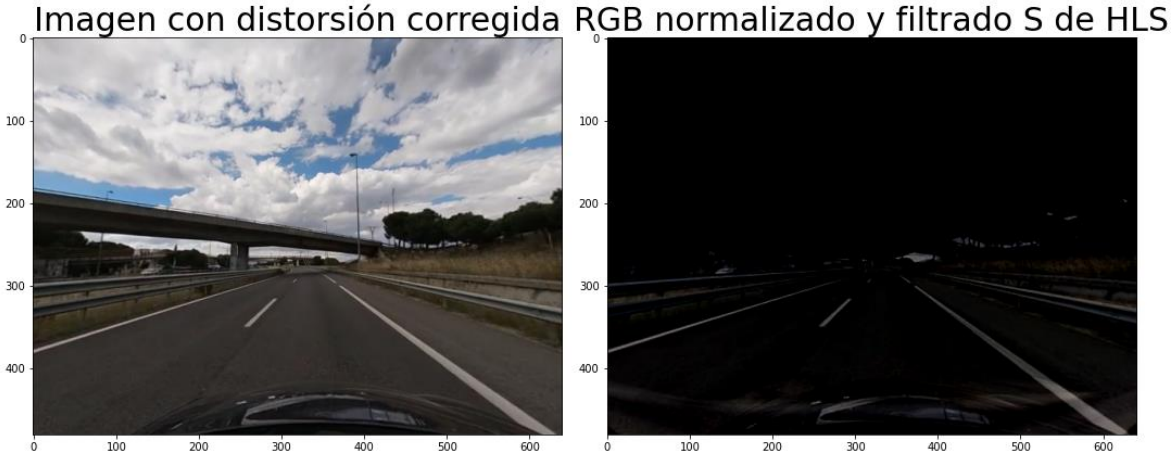


Figura 3-7: Ejemplo normalización de canales RGB y filtrado de canal S en HSL

3.4 Etapa 3: Aplicación de umbrales Otsu y Sobel

En la tercera etapa, se procede a la umbralización de los fotogramas para poder segmentar la imagen. Tras varios ensayos para determinar el método de segmentación idóneo a utilizar, se llegó a la conclusión que los mejores resultados eran producidos mediante una combinación de los umbrales Otsu y Sobel.

Esta etapa estará formada por tres partes. En la primera parte, se transforma la imagen de los canales RGB a una escala de grises. Con la imagen en escala de grises, se le aplica la técnica de desenfoque Gaussiano usando un núcleo de 3x3, donde se convoluciona la imagen con el núcleo especificado.

En la segunda parte, volveremos a tomar la imagen previa de la primera parte, antes de operar con ella, y la transformaremos al espacio de color HSL. Con el canal L, aplicaremos el umbral Sobel que, con nuestros valores, el núcleo de la convolución corresponde a:

$$\begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix}$$

Con esta operación conseguimos evitar las líneas horizontales en nuestro proceso de segmentación. En este paso también normalizaremos los valores del umbral Sobel y filtraremos cualquier valor que no se encuentren entre los valores 20 y 100, con la finalidad de descartar ruido creado durante el proceso de umbralización.

En la tercera parte, aplicaremos la función lógica OR al primer y segundo paso, obteniendo así una segmentación más clara.

A continuación, se muestran imágenes de los resultados de los pasos de la segmentación, tanto con una imagen sin normalizar como con una imagen normalizada.

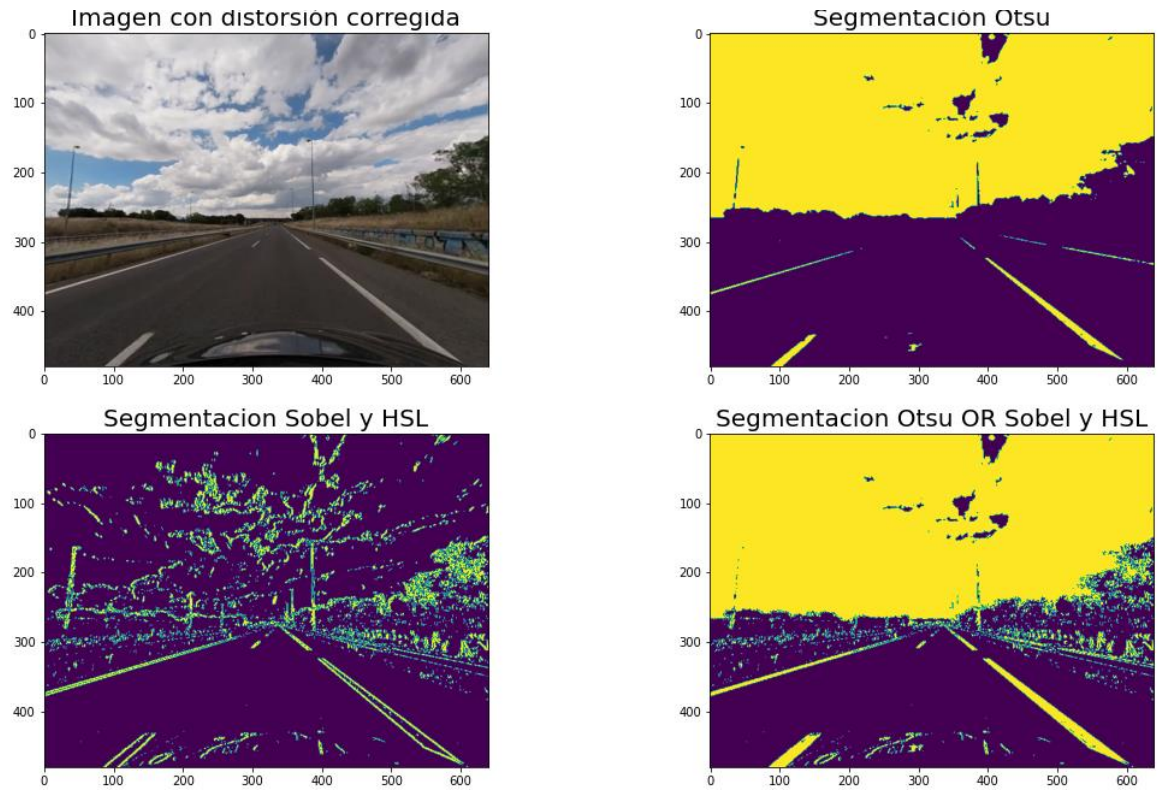


Figura 3-8: Ejemplo segmentación sobre imagen no normalizada

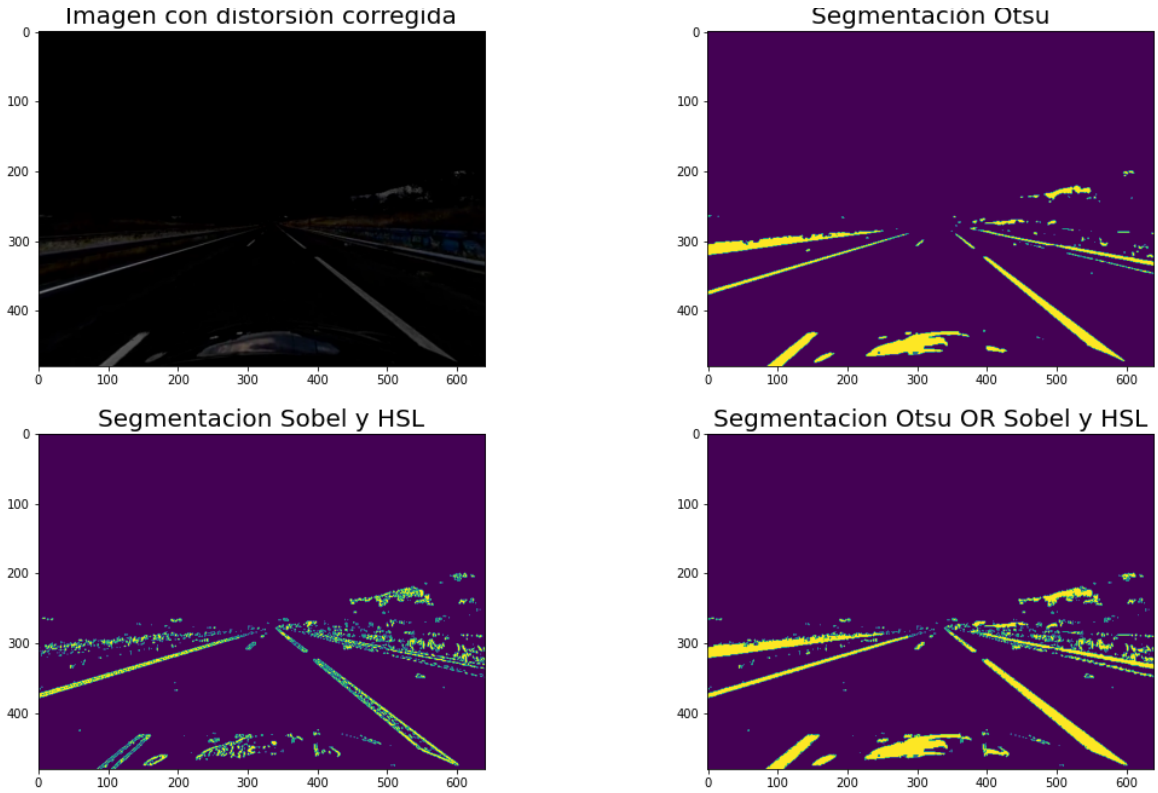


Figura 3-9: Ejemplo segmentación sobre imagen normalizada

3.5 Etapa 4: Identificación de la región de interés

Para la identificación de la región de interés, mediante el análisis de los vídeos se ha determinado la zona dónde se pueden encontrar las marcas de trazado de la carretera la mayoría del tiempo. Se concluye que esta región es bastante estándar, puesto que la cámara está fijada en el salpicadero y techo del coche. Por lo tanto, se opta por definir los píxeles de la región de interés de una forma estática con respecto a los valores de altura y anchura de la imagen.

Las coordenadas de lo píxeles de la figura que denota la región de interés son:

[(165,425), (300, 290), (360, 290), (580, 425)]

El primero de ellos sobre una imagen normalizada:

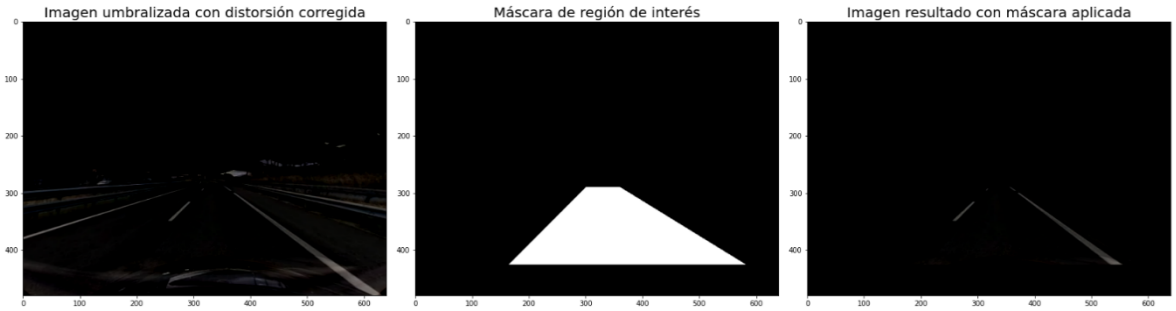


Figura 3-10: Ejemplo aplicación de máscara a imagen no normalizada para la extracción de la región de interés

El segundo sobre una imagen umbralizada:

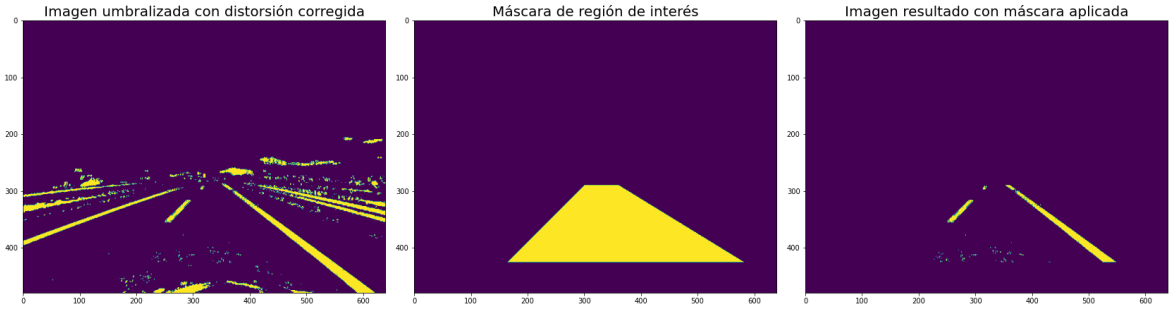


Figura 3-11: Ejemplo aplicación de máscara a imagen umbralizada para la extracción de la región de interés

3.6 Etapa 5: Transformación de perspectiva

Una vez acotada la región de interés, se realiza una transformación de perspectiva. El objeto de esta transformación es simplificar la detección de las marcas del trazado de la carretera, puesto que la imagen adopta una perspectiva de vista de pájaro. Como se explicó en el Capítulo 2.2, la transformación de perspectiva facilita la detección de objetos o regiones al tratarse de una imagen en dos dimensiones.

Para la transformación de perspectiva se tomarán los puntos de la imagen que queremos transformar y se calcularán los puntos a los que queremos que la imagen quede transformada.

En el caso de este trabajo, los puntos originales serán equivalentes a los de la región de interés:

[(165,425), (300, 290), (360, 290), (580, 425)]

Y los puntos elegidos como destino serán:

[(100,430), (200, 135), (500, 135), (580, 430)]

Con estos pares de puntos, se obtiene la matriz de transformación M con la siguiente ecuación:

$$\begin{pmatrix} t_i x'_i \\ t_i y'_i \\ t_i \end{pmatrix} = M * \begin{pmatrix} x_i \\ y_i \\ 1 \end{pmatrix}$$

Siendo M una matriz de tres por tres, los puntos destino (x'_i, y'_i) y los puntos origen (x_i, y_i), para i = 0,1,2,3.

Una vez calculada la matriz de transformación M, se pueden calcular los puntos de transformación de la siguiente manera:

$$dst(x,y) = src(\frac{M_{11}x + M_{12}y + M_{13}}{M_{31}x + M_{32}y + M_{33}}, \frac{M_{21}x + M_{22}y + M_{23}}{M_{31}x + M_{32}y + M_{33}})$$

Siendo src los puntos de origen y dst los puntos destino de la transformación.

A continuación, se muestran dos ejemplos de transformación de perspectiva sobre la misma imagen. El primero sobre la imagen umbralizada y el segundo sobre la imagen normalizada.

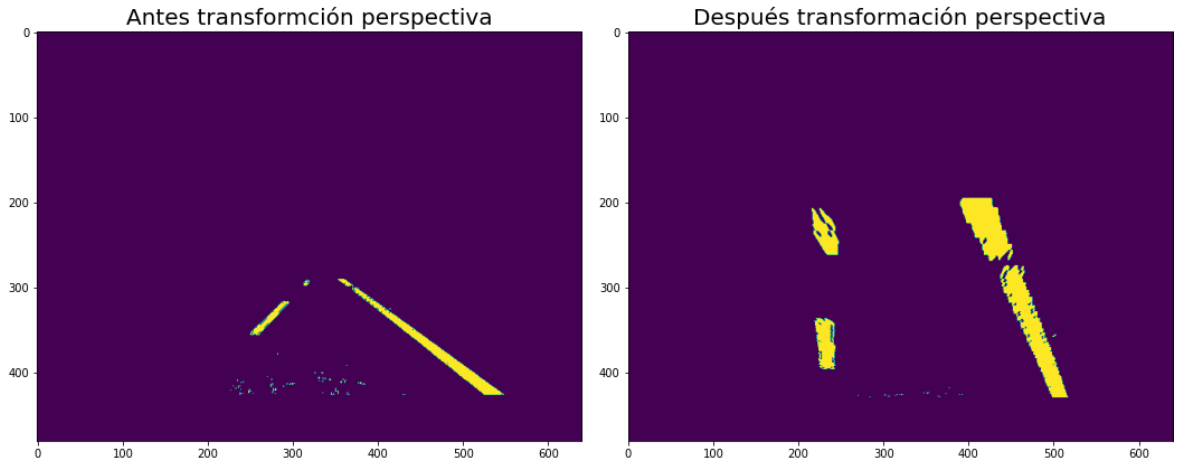


Figura 3-12: Ejemplo transformación de perspectiva sobre imagen umbralizada

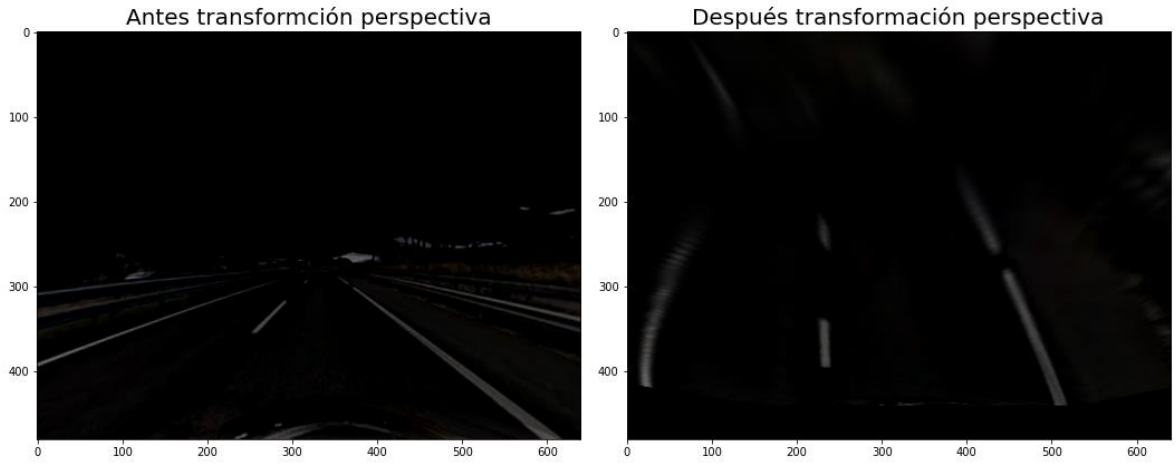


Figura 3-13: Ejemplo transformación de perspectiva sobre imagen normalizada

3.7 Etapa 6: Detección de marcas de trazado de la carretera mediante la aplicación del algoritmo SWT

Con la imagen transformada y habiendo aplicado el umbral a la imagen normalizada, podemos obtener mediante un histograma un primer indicio de dónde tratar de buscar en la imagen para detectar las líneas del trazado de la carretera.

En el siguiente histograma, se puede observar la dispersión de los píxeles en el eje de la anchura, pudiendo obtener una estimación de por dónde empezar a buscar con el algoritmo SWT en la imagen. Esta estimación la realizamos maximizando la media de los cien píxeles tanto de la parte izquierda como de la derecha del histograma.

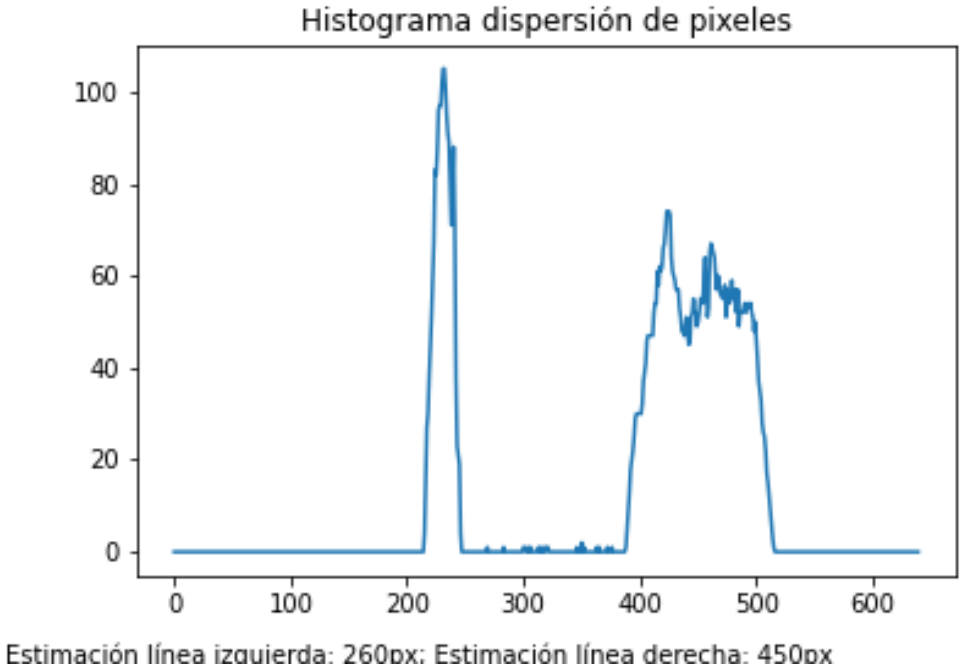


Figura 3-14: Histograma de dispersión de píxeles de una imagen umbralizada y transformada

Con estas estimaciones, podemos aplicar el algoritmo SWT. Se dividirá la imagen verticalmente en el número de ventanas que queramos emplear para el algoritmo, en nuestro caso 9 ventanas. Cada una de ellas tendrá dos ventanas interiores que buscarán en un margen de 70 píxeles en este caso concreto. Si dentro de cada una de éstas se encontrase un número de píxeles superior a un umbral definido -en el caso de este algoritmo 40 píxeles- se marcará como encontrada la línea de carretera en la ventana interior, guardando las coordenadas de los píxeles y continuando con la iteración de la siguiente ventana.

Para el resto de las iteraciones sobre las ventanas restantes, se procederá al igual que en la primera de ellas. Se dividirá en dos ventanas interiores, y se buscará una densidad de píxeles que supere un umbral predefinido dentro de un margen predeterminado.

A continuación, se expone una imagen con el desarrollo de cada iteración del algoritmo. Se puede observar cómo va incrementando el número de ventanas interiores según aumentan las iteraciones del algoritmo, de modo que se delimitan las líneas de la carretera.

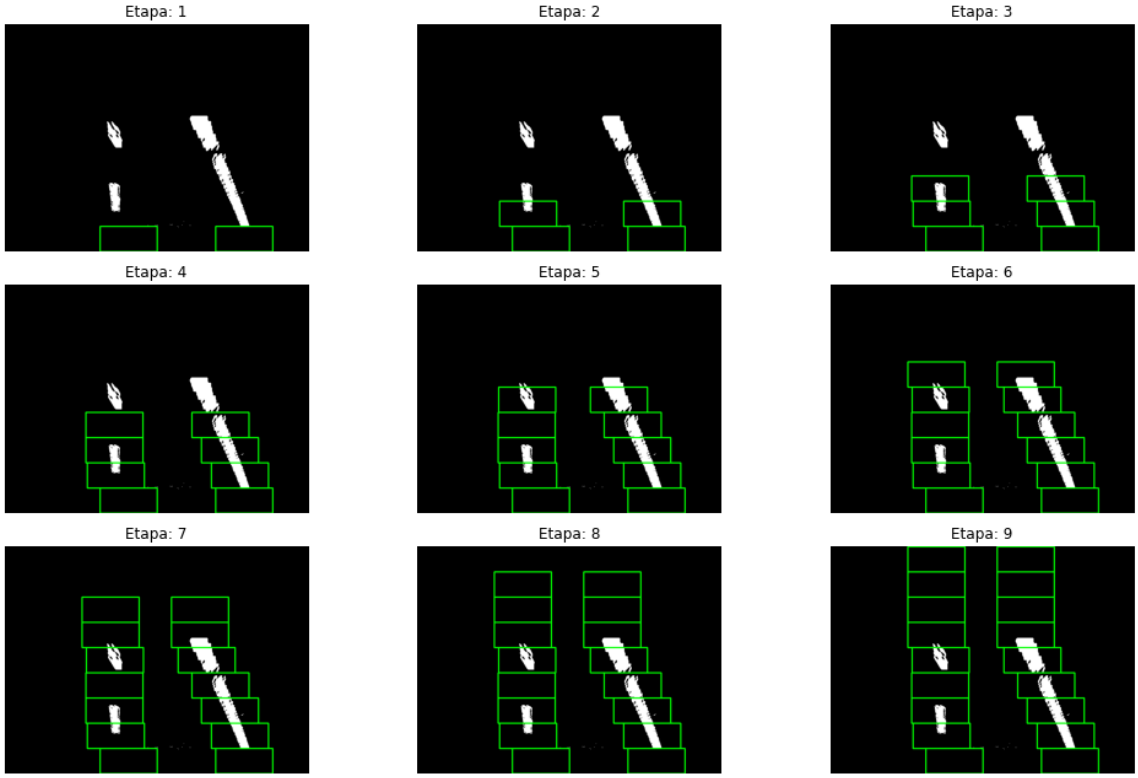
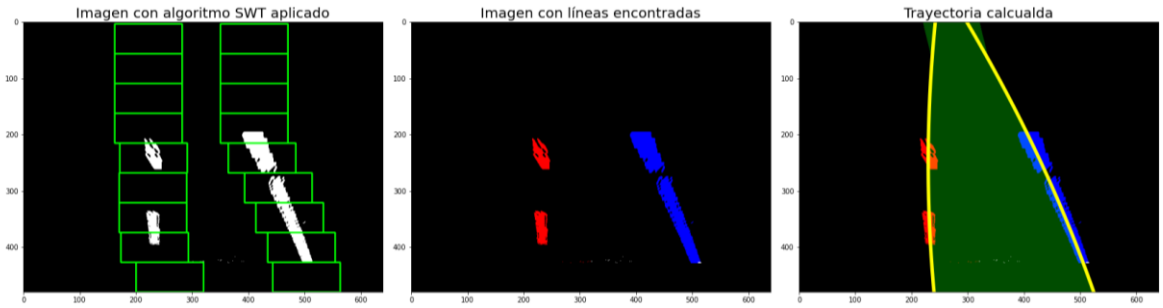


Figura 3-15: Ejemplo iteraciones del algoritmo SWT

3.8 Etapa 7: Cálculo de la trayectoria del vehículo

Con el resultado del algoritmo SWT, podemos estimar la ecuación determinada por cada una de las líneas de la carretera, obteniendo las coordenadas de las ventanas interiores obtenidas. Habiendo obtenido estas ecuaciones, podemos calcular la trayectoria deseada para el vehículo.



Estimación ecuación línea izquierda: 0.000198 x^2 + -0.099602 x + 242.309344 px
Estimación ecuación línea derecha: -0.000254 x^2 + 0.592008 x + 298.749171 px

Figura 3-16: Ejemplo de trayectoria calculada a partir del resultado del algoritmo SWT

Una vez obtenida la trayectoria, se aplica una transformación de perspectiva inversa como en la etapa cinco, pero intercambiando los puntos de origen y destino a la hora de obtener la matriz de transformación.

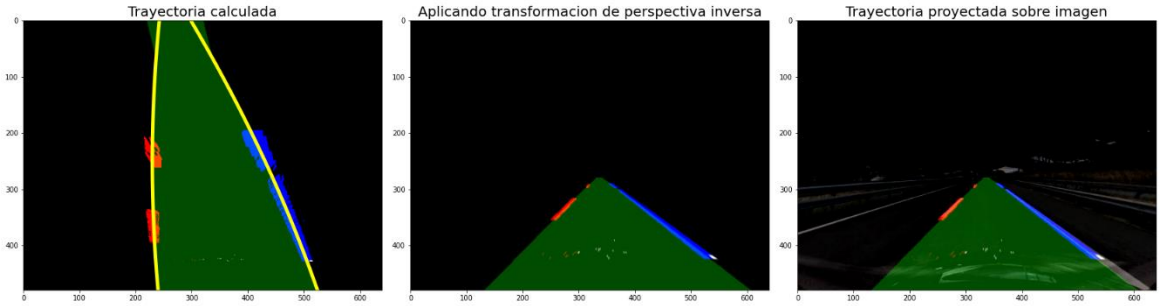


Figura 3-17: Ejemplo transformación inversa de perspectiva proyectada sobre el fotograma del vídeo normalizado.

4 Pruebas y resultados

4.1 Pruebas

Inicialmente se experimentó con la desviación estándar sobre el total de fotogramas de un vídeo, tratando de delimitar una zona de la calzada donde pudiesen estar situadas las líneas que delimitan el trazado de ésta, aunque no se pudo sacar ninguna conclusión aplicable al algoritmo usado.

Se comenzó calculando la desviación estándar sobre las imágenes en blanco y negro, sin obtener ningún resultado aparentemente satisfactorio. A continuación, se trató de calcular también la desviación estándar sobre cada canal RGB por separado para más tarde juntar las desviaciones en una única imagen, aunque tampoco se obtuvieron resultados concluyentes.

Se adjuntan los resultados obtenidos de los cálculos comentados:

Figura 4-1: Desviación estándar sobre los fotogramas de un vídeo en blanco y negro

Se realizaron pruebas sobre el espacio de color YUV, donde operando sobre el canal Y, y filtrando la imagen con la media de ese canal para luego invertirla, se conseguían buenos resultados para segmentar las líneas del trazado de la carretera. Esta prueba se tuvo que descartar puesto que al realizar la normalización RGB, no se producían los mismos resultados que antes de normalizar la imagen.

Se adjunta un ejemplo de los experimentos sobre imágenes sin normalizar e imágenes normalizadas.

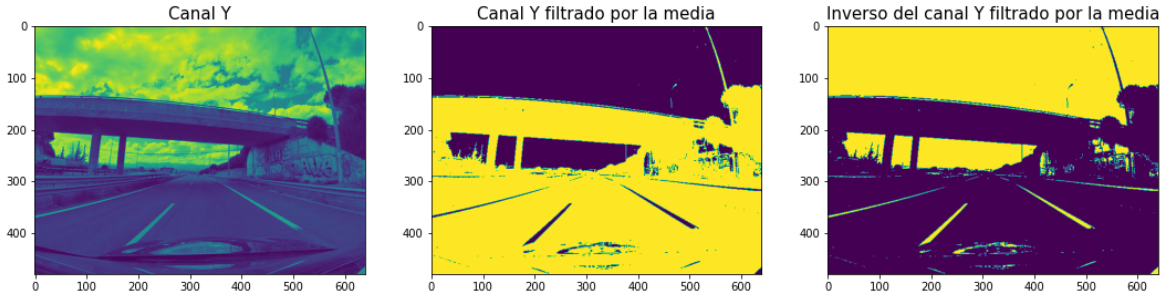


Figura 4-2: Experimento segmentación sobre canal Y del espacio de colores YUV

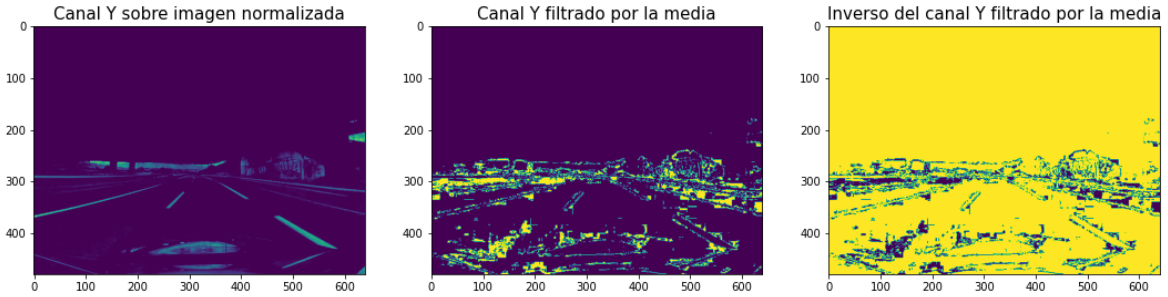


Figura 4-3: Demostración fallo de experimento de segmentación sobre canal Y del espacio de colores YUV sobre una imagen con canales RGB normalizados

Se experimentó también con umbrales adaptativos, tanto el método Gaussiano como el método de la media de la imagen. En este caso, no se obtuvieron resultados óptimos debido a que estos métodos mencionados añaden bastante ruido en la periferia de las marcas viales comparado con la metodología utilizada finalmente en el sistema. A continuación, se adjunta un ejemplo de las pruebas realizadas con los umbrales adaptativos y una comparación con el umbral utilizado en el sistema.

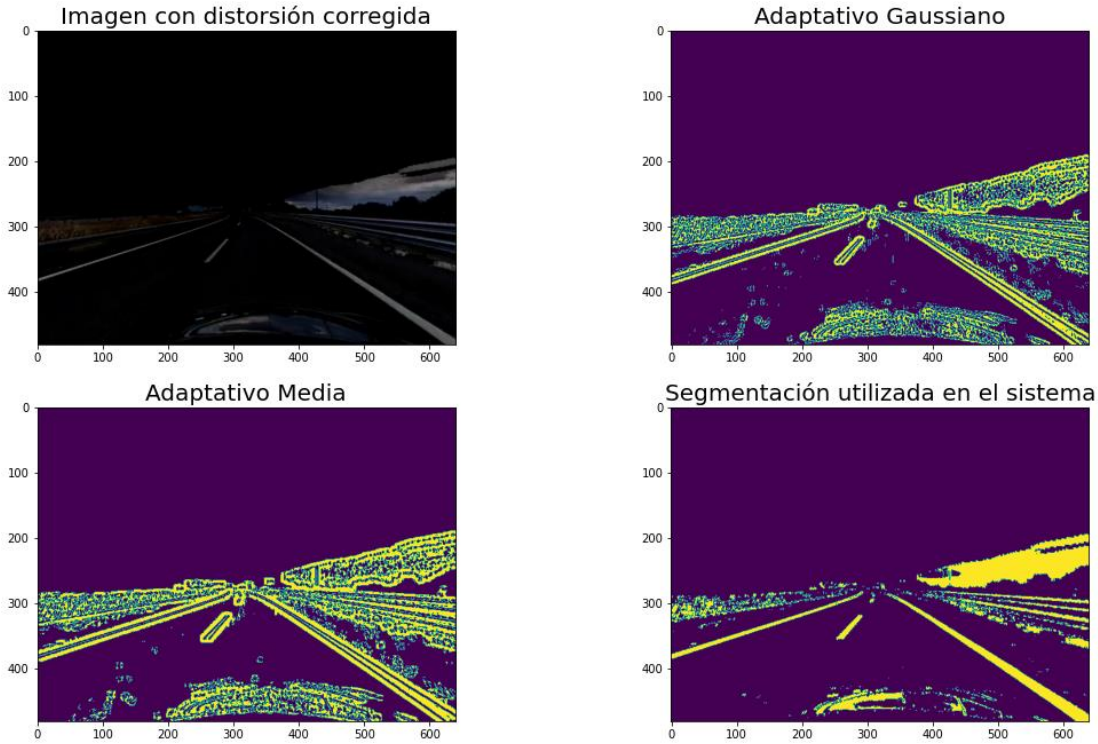


Figura 4-4: Experimento umbrales adaptativos

Como se ha comentado en la sección 5.1, originalmente se diseñó el sistema capturando las imágenes desde el salpicadero del vehículo. Esta aproximación se ha descartado tras varias pruebas porque a la hora de manipular los fotogramas, la menor altura de la posición de la cámara respecto a la opción de colocarla en el techo del vehículo reduce la profundidad de la imagen, haciendo más difícil la detección de marcas de trazado de la carretera.

Se adjuntan unos ejemplos de fotogramas extraídos de vídeos tomados desde el salpicadero del coche donde se producen errores debidos a lo comentado en el párrafo anterior. Estos ejemplos han sido aceptados por un validador, como se explica en el siguiente apartado.

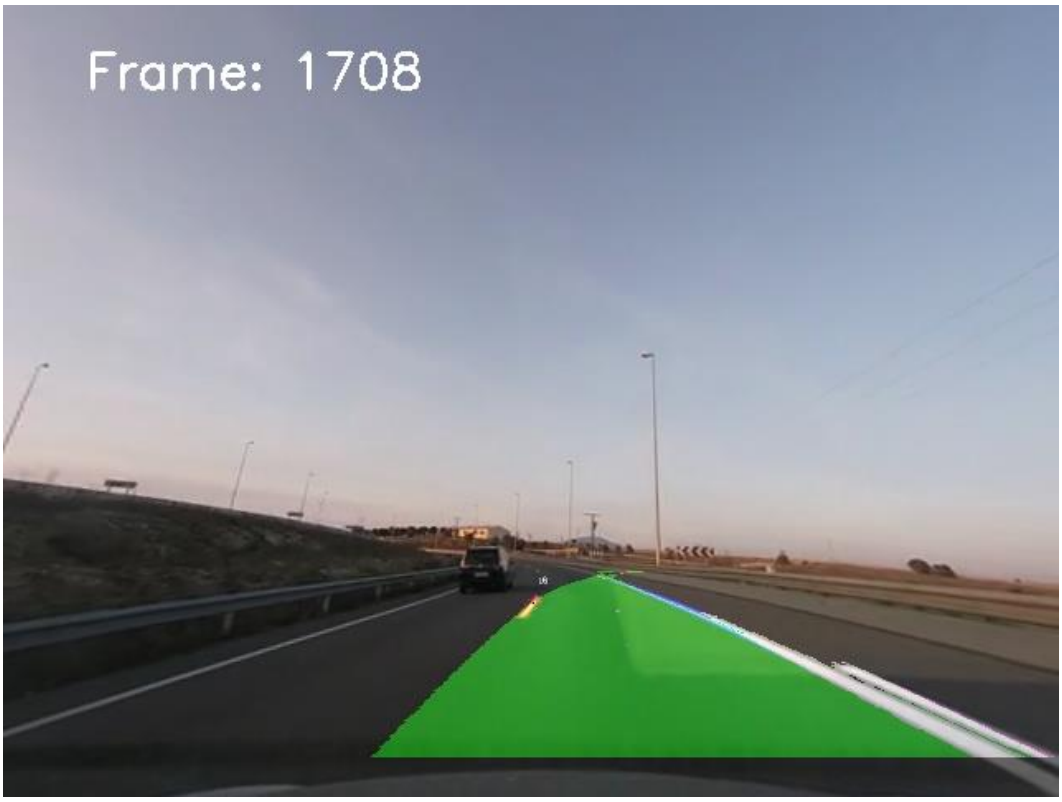


Figura 4-5: Ejemplo fallo de predicción de trayectoria debido a la falta de profundidad en la imagen

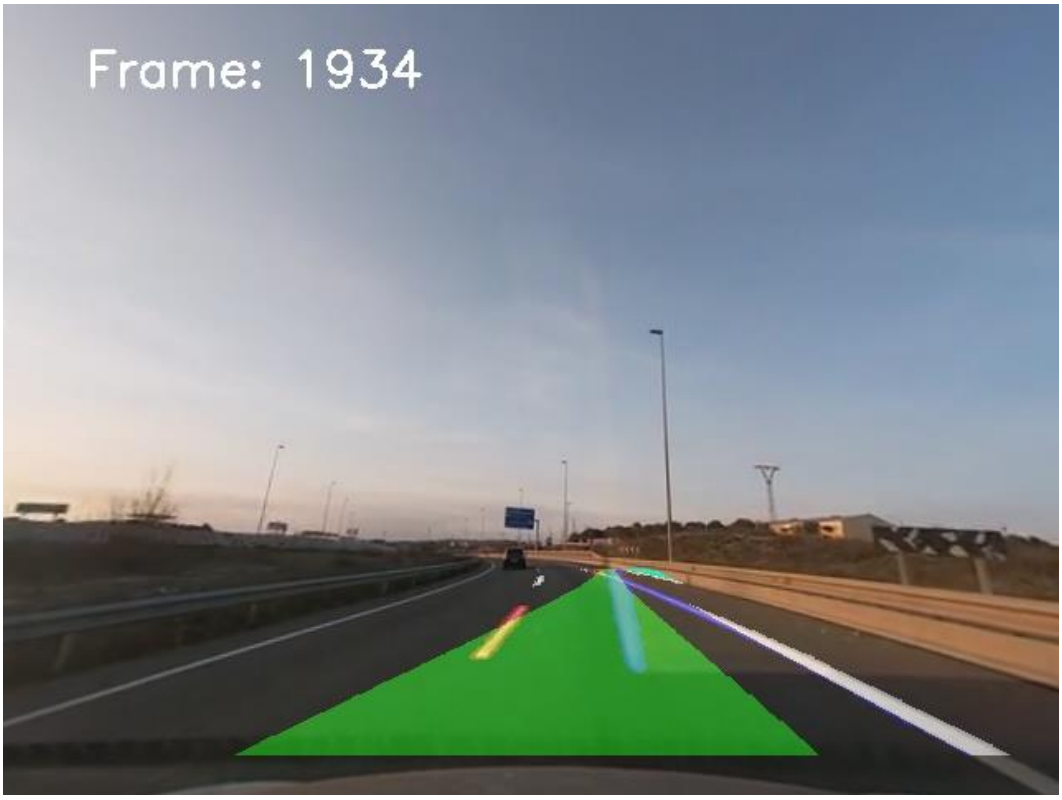


Figura 4-6: Ejemplo fallo de predicción por falta de profundidad en la imagen y por variables de entorno de grabación como reflejos en el parabrisas

A continuación, se adjuntan también ejemplos donde el cálculo de la trayectoria del vehículo ha sido satisfactorio desde el salpicadero de éste, también aceptadas por el validador.

Figura 4-7: Ejemplo predicción satisfactoria de recta tomada desde el salpicadero

Figura 4-8: Ejemplo predicción satisfactoria con líneas discontinuas

Se ha experimentado con el algoritmo sobre imágenes nocturnas en las mismas vías que las imágenes mostradas anteriormente. A la hora de preprocesar las imágenes nocturnas, la normalización y la umbralización se ha modificado debido a que la explicada en el documento producía errores por los reflejos producidos por las farolas. Se ha quitado la normalización RGB y HSL, y la umbralización por el método de Otsu, dejando únicamente la umbralización por el método de Sobel. Al reducir la umbralización, el algoritmo produce resultados menos acertados, teniendo que realizar más investigación en este ámbito.

A continuación, se muestran algunos de los resultados producidos con imágenes nocturnas, mostrando alguno de los errores y aciertos de la trayectoria.

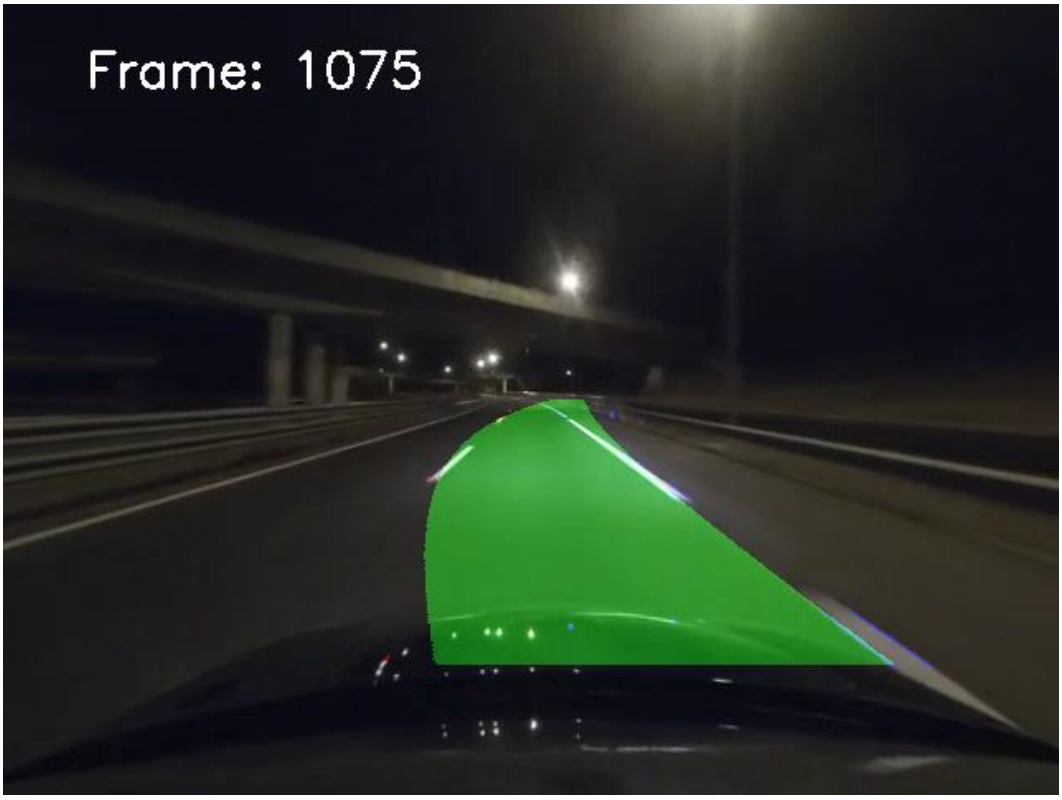


Figura 4-9: Demostración fallo predicción de trayectoria en imagen nocturna

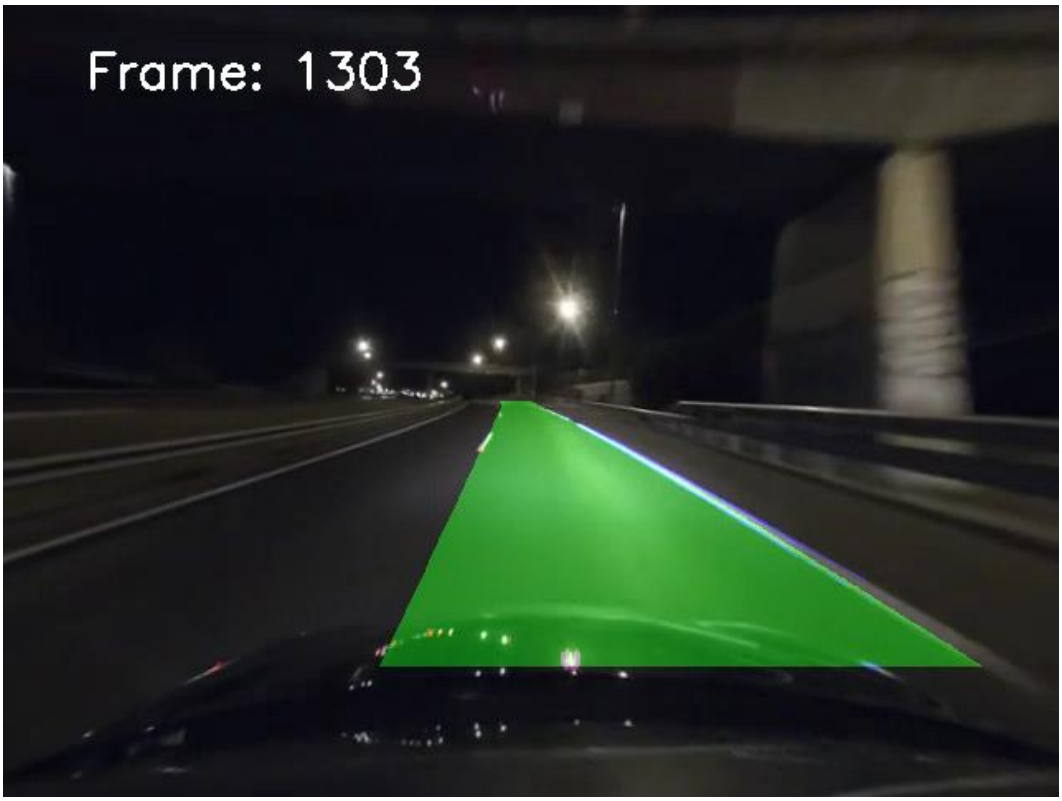


Figura 4-10: Ejemplo de predicción correcta en imagen nocturna

Figura 4-11: Ejemplo predicción correcta en imagen nocturna tras cambio de carril

4.2 Resultados

En este apartado de la memoria se analizan los resultados obtenidos por parte de la aplicación del sistema a un vídeo tomado desde el techo y salpicadero del vehículo, aplicando el proceso de tratamiento de las imágenes explicado detalladamente en el Capítulo 3.

Se ha demostrado prácticamente que el enfoque de este trabajo ha sido correcto, produciendo resultados muy positivos durante la ejecución del algoritmo. La normalización RGB y el filtrado HSL implementados resaltan las marcas del trazado de la carretera, proporcionando mejores resultados que los obtenidos sin aplicar este preprocesado.

El mecanismo de búsqueda de la mayor densidad de píxeles, realizado en este trabajo, perfecciona la ejecución del sistema, produciendo una cantidad menor de falsos positivos debidos a marcas producidas por destellos, reflejos o imperfecciones en la carretera. Esta implementación facilita posteriormente la ejecución del algoritmo SWT, produciendo mejores resultados en las trayectorias calculadas.

A continuación, se mostrarán algunos casos a destacar de resultados producidos por el sistema:

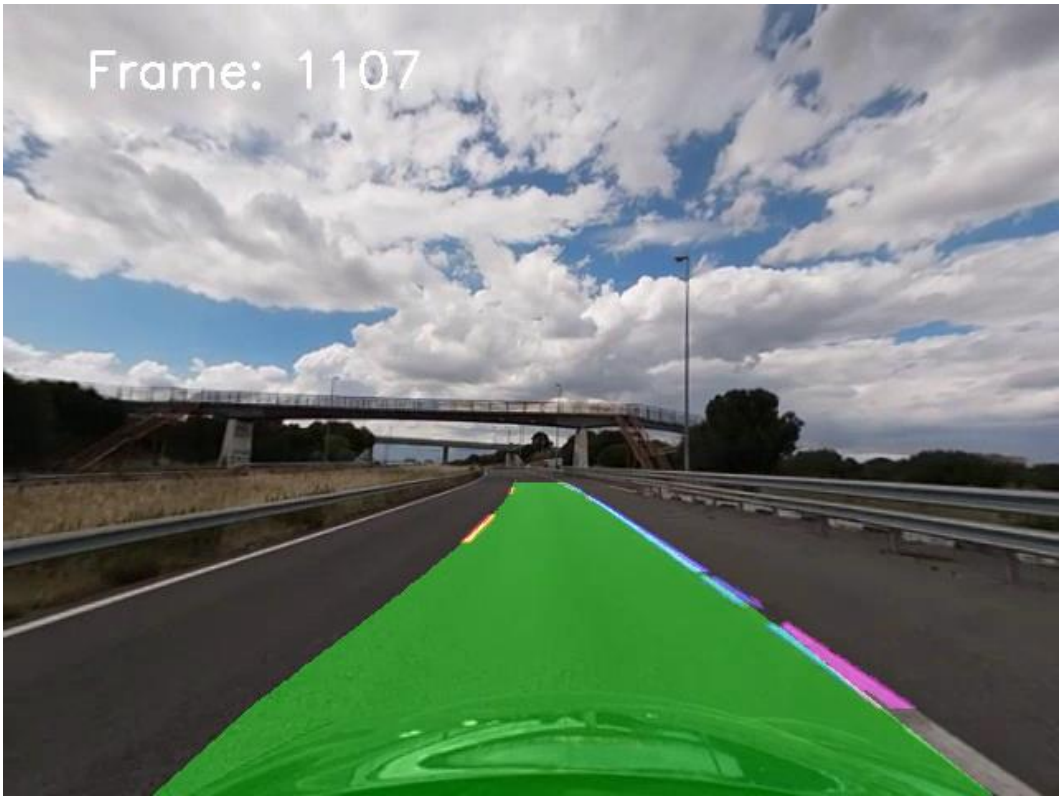


Figura 4-12: Ejemplo predicción de trayectoria curva a izquierdas

Figura 4-13: Ejemplo de predicción de curva a izquierda con marcas de trazado no deseadas

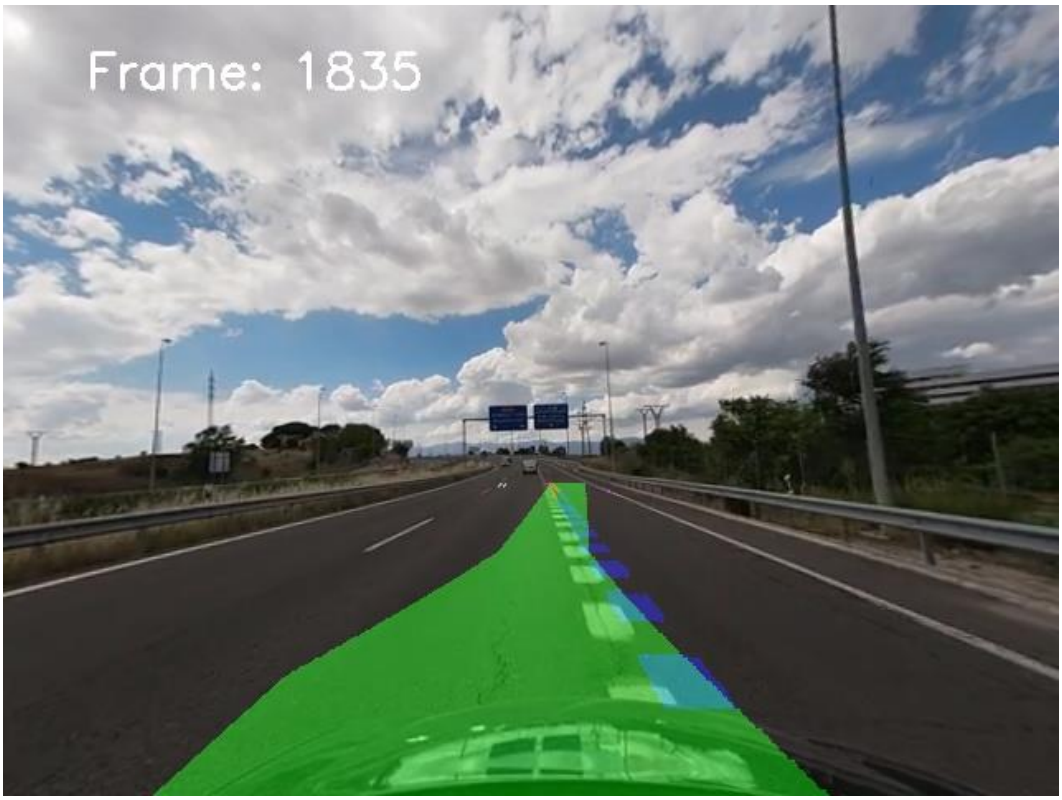


Figura 4-14: Ejemplo de predicción durante un cambio de carril a derechas

Figura 4-15: Ejemplo de predicción durante un cambio de carril desde un carril situado a la izquierda

Figura 4-16: Ejemplo de predicción después de un cambio de carril

El sistema se puede ver afectado por imágenes que contengan objetos no deseados en la trayectoria de la carretera, u objetos que no se han contemplado, como marcas viales situadas en la calzada. Para corregir estos problemas, se ha implementado un validador de trayectorias.

Este validador calcula el porcentaje de trayectoria coincidente entre dos fotogramas subsiguientes. Si este porcentaje es mayor a un umbral definido previamente, en nuestro caso 95%, se considerará que el fotograma actual tiene una trayectoria adecuada respecto al fotograma anterior, y entonces se acepta. Si la relación entre ambas trayectorias es menor al umbral definido, se rechaza la trayectoria calculada para el fotograma actual, y se sustituye por la calculada para el fotograma anterior. Si se han rechazado las trayectorias de cinco o más fotogramas seguidos, entonces se evalúa si se ha podido cometer un error rechazando los fotogramas. Para evaluar este error, se recogen los cuatro fotogramas anteriores y el actual, y se aplica el validador entre esos cinco fotogramas. Si el porcentaje de trayectoria coincidente entre esos fotogramas es mayor al del umbral definido, se acepta la trayectoria calculada para el actual, y se reinicia el contador de errores. En el caso de que la evaluación no dé un resultado positivo, entonces se aumenta el contador de errores y se mantiene la prolongación del último fotograma que se ha validado positivamente.

La fórmula seguida para calcular el porcentaje de similitud entre las trayectorias de dos imágenes diferentes, A y B, es la siguiente:

$$\frac{(número\ de\ pixeles\ comunes\ con\ valor\ mayor\ que\ 0\ en\ A\ y\ B) * 2}{número\ píxeles\ con\ valor\ mayor\ que\ 0\ de\ A + \ número\ píxeles\ con\ valor\ mayor\ que\ 0\ de\ B}$$

Este validador diseñado e implementado en este trabajo, permite al sistema obtener mejores resultados. Los resultados validados son más consistentes que las pruebas realizadas sin el validador, mejorando notoriamente la trayectoria calculada cuando el sistema falla y predice una trayectoria que no es totalmente precisa.

A continuación, se muestra un fotograma donde debido a la sombra de un puente, el umbral de la imagen no es el deseado, y mediante el evaluador, la trayectoria del vehículo es corregida a una trayectoria anterior.

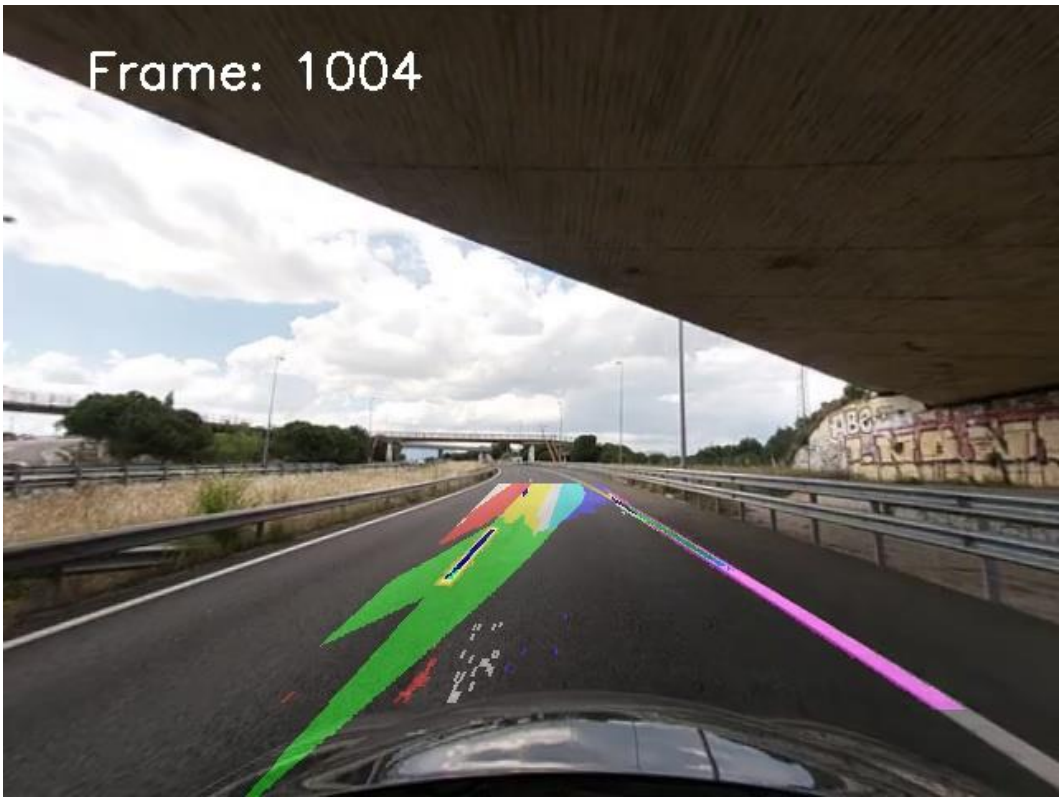


Figura 4-17: Demostración fallo de predicción de trayectoria por sombras en el trazado

Figura 4-18: Demostración corrección de trayectoria con sombras en el trazado mediante evaluador

Otro ejemplo donde la actuación del validador tiene efecto, es cuando existen deterioro en la carretera o marcas no deseadas. Como se puede ver en las siguientes imágenes, el validador ignora el fotograma donde, debido al mal estado de la calzada, se produce un mayor ruido en la umbralización de la imagen, prolongando la trayectoria calculada para el fotograma anterior.

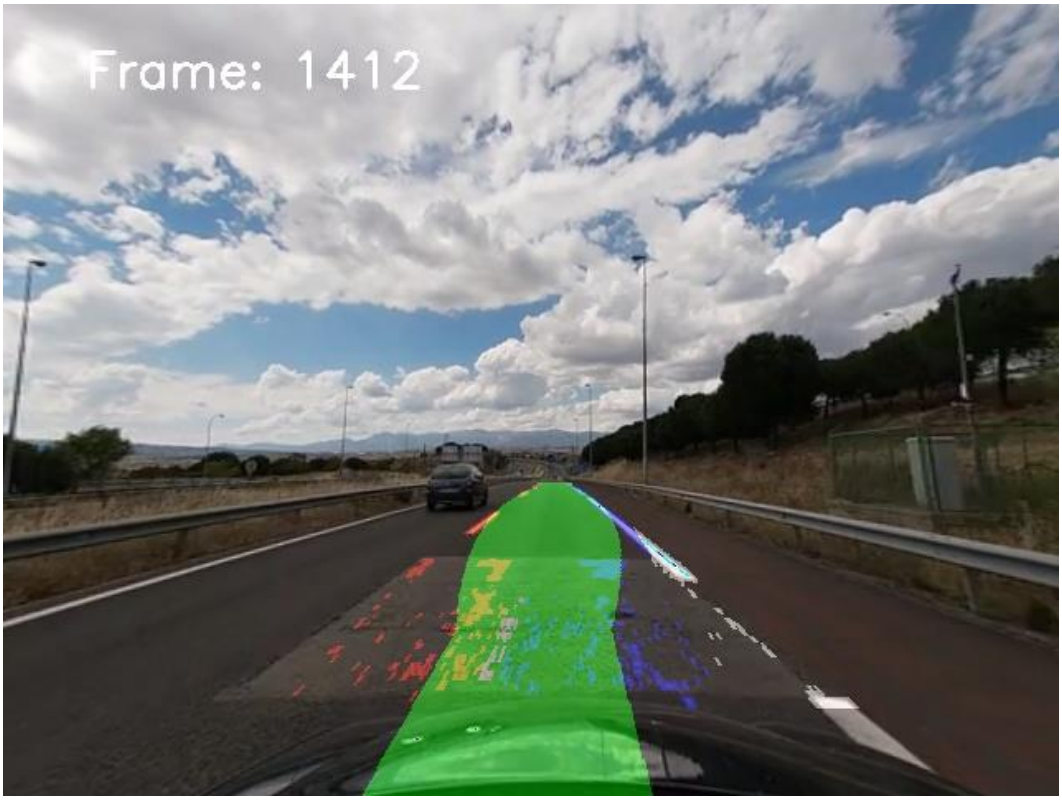


Figura 4-19: Demostración fallo de predicción de trayectoria por deterioro de la carretera

Figura 4-20: Demostración corrección de trayectoria con deterioro en la carretera mediante evaluador

5 Conclusiones y trabajo futuro

5.1 Conclusiones

Tras la finalización de este trabajo, se pueden extraer las siguientes conclusiones:

1. La velocidad de ejecución y procesamiento alcanzados mediante la aplicación de las técnicas desarrolladas al algoritmo es aproximadamente de 30 fotogramas por segundo, valor que se corresponde con el objetivo marcado previo a la realización de este trabajo.

2. El preprocesado de imagen desarrollado resulta vital para detectar mejor las líneas y reducir el número de reflejos constantes producidos en el parabrisas y en el capó del vehículo.

3. La segmentación y la umbralización están condicionadas por el preprocesado de la imagen, y la combinación de éstas ideada en este trabajo son necesarias para facilitar la ejecución del algoritmo y para el mejor funcionamiento de éste.

4. La transformación de perspectiva constituye una técnica de tratamiento de imágenes fundamental a la hora del cálculo de trayectorias de vehículos sin el uso de sensores, debido a la profundidad que ésta aporta en los cálculos.

5. El algoritmo SWT resulta un algoritmo muy adecuado para delimitar líneas de carretera en una imagen.

6. El método elaborado para la búsqueda de mayor densidad de píxeles en un rango predefinido para realizar una hipótesis inicial de la posición de las líneas aporta robustez al sistema, reduciendo la aparición de falsos positivos debidos a marcas producidas por destellos, reflejos o imperfecciones en la carretera.

7. El validador desarrollado aporta un mecanismo necesario para la corrección de pequeños errores que surjan durante el cálculo de la trayectoria, causados por elementos fortuitos, tales como deterioros de la carretera, sombras no deseadas y otras variables ajenas al sistema.

8. Se ha demostrado un rendimiento mayor del sistema con las imágenes tomadas desde el techo del vehículo que con las imágenes tomadas desde el salpicadero para el conjunto de datos creado.

9. Con imágenes nocturnas, el sistema debe prescindir del preprocesado y cambiar la umbralización de las imágenes para producir unos resultados aceptables, aunque no alcanza la consistencia de los resultados producidos con imágenes diurnas.

10. El futuro de la conducción va ligado a la tecnología, de modo que parece probable que se alcance una completa automatización, sin la necesidad de la supervisión de un ser humano, y con una tasa de fallo menor que la inherente a las distracciones y errores de los conductores humanos.

Como añadido y elemento de contraste, en las fases finales de la realización de este Trabajo de Fin de Grado se ha encontrado en internet el artículo escrito por Raja Muthalagu, Anudeepsekhar Bolimera y V. Kalaichelvi [28] que será publicado en Julio de 2020 según

información de la publicación online (https://www.sciencedirect.com/science/article/pii/S0045790620305085). En él, se presentan dos versiones de algoritmos de cálculos de trayectoria: una minimalista y otra más avanzada y parecida a la realizada en este TFG. En la avanzada, realizan transformaciones a los espacios HLS y LAB y aplican umbralización Sobel y umbralización binaria adaptativa para el preprocesamiento de imagen, para después realizar la detección de líneas de la carretera mediante la aplicación del algoritmo SWT con el fin de calcular la trayectoria del vehículo. Tras su lectura, a continuación, se muestran las conclusiones de la comparativa, fortalezas y debilidades entre el enfoque de este Trabajo de Fin de Grado y el artículo en cuestión:

a. Como fortaleza de este TFG respecto al artículo, realizar previamente la combinación de normalización RGB y filtrado HSL permite resaltar las marcas del trazado vial antes de segmentar, lo cual produce mejores resultados en la segmentación de las líneas.

b. Otra ventaja de este TFG es que la aplicación de los umbrales Otsu y Sobel produce mejores resultados en la umbralización de la imagen que el umbral Sobel, como se ha demostrado en el Capítulo 3.4 de este trabajo.

c. En el artículo, se genera una hipótesis de la posición de las líneas de carretera buscando los valores máximos del histograma. El método empleado en este TFG añade robustez, al implementar un método para la búsqueda de las regiones con mayor densidad de píxeles en un ancho de 100 píxeles, en vez de búsqueda por máximos puntuales, omitiendo así posibles fallos en la aplicación del algoritmo SWT.

d. En este TFG se ha creado un evaluador de fotogramas para la validación de la trayectoria, mientras que en [28] se ha usado la métrica RMSE. El evaluador permite corregir en el momento las trayectorias identificadas como erróneas, mientras que el RMSE aporta datos de efectividad de rendimiento del algoritmo. Ambos mecanismos son acumulables.

5.2 Trabajo futuro

Las opciones de trabajo futuras que se consideran, para la mejora de este sistema se enumeran a continuación:

- Detección de marcas viales diferentes de las que delimitan el trazado, que puedan afectar a la predicción de la trayectoria.
- Generar un vídeo etiquetado manualmente para medir el porcentaje de acierto del sistema.
- Ampliación del campo de visión del coche y robustez del sistema, introduciendo más cámaras para mejorar la detección de información y contemplar el caso de indisponibilidad (por ejemplo, ante ensuciamiento repentino de la lente) de la cámara.
- Inclusión de sensores que permitan aumentar la precisión de la detección de las marcas de trazado, lo cual quizás permitiría reducir el coste computacional al reducir el preprocesado de imagen y de segmentación.
- Realizar el cálculo de preprocesado de las imágenes teniendo en cuenta un número predefinido de los últimos fotogramas tomados, para que el sistema resulte funcional en tiempo real.

- Posible traslación de los datos generados por el sistema como datos etiquetados para su implementación en redes neuronales, con la finalidad de dar una aproximación diferente al problema y comparar los resultados obtenidos con los de este sistema.
- Incremento de los datos y variables de entorno proporcionados al sistema, como pueden ser velocidad del vehículo, posición del volante y radio de giro, hora diurna, o posición GPS para refinar los cálculos y enriquecer la información de salida.
- Añadir la detección de vehículos u objetos circundantes al sistema para tenerlos en cuenta a la hora de calcular la trayectoria deseada.

Referencias

[1] B. T. Nugraha, S. Su and Fahmizal, "Towards self-driving car using convolutional neural network and road lane detector,", 2017 2nd International Conference on Automation, Cognitive Science, Optics, Micro Electro--Mechanical System, and Information Technology (ICACOMIT), Jakarta, 2017, pp. 65-69, doi: 10.1109/ICACOMIT.2017.8253388.

[2] Mariusz Bojarski, Davide Del Testa, Daniel Dworakowski, Bernhard Firner, Beat Flepp, Prasoon Goyal, Lawrence D. Jackel, Mathew Monfort, Urs Muller, Jiakai Zhang, Xin Zhang, Jake Zhao, Karol Zieba, "End to End Learning for Self-Driving Cars", arXiv: 1604.07316 [cs] 2016

[3] Nan, Zhixiong and Wei, Ping and Xu, Linhai and Zheng, Nanning, "Efficient Lane Boundary Detection with Spatial-Temporal Knowledge Filtering", Sensors, Article 1276, Number 8, Volume 16, 2016, ISSN 1424-8220, doi: 10.3390/s16081276

[4] Yue Wang, Eam Khwang Teoh, Dinggang Shen, "Lane detection and tracking using B-Snake", Image and Vision Computing, Volume 22, Issue 4, 2004, Pages 269-280, ISSN 0262-8856,

[5] G. Walberg, "GEOMETRIC TRANSFORMATION TECHNIQUES FOR DIGITAL IMAGES: A SURVEY", December 1988, Columbia University, Technical Report CUCS-390-88.

[6] A. Bevilacqua, A. Gherardi and L. Carozza, "Automatic Perspective Camera Calibration Based on an Incomplete Set of Chessboard Markers," 2008 Sixth Indian Conference on Computer Vision, Graphics & Image Processing, Bhubaneswar, 2008, pp. 126-133, doi: 10.1109/ICVGIP.2008.10.

[7] Heckbert, Paul. (1986). "Survey Of Texture Mapping." Computer Graphics and Applications, IEEE. 6. 56-67. 10.1109/MCG.1986.276672.

[8] J. Gwak, J. Jung, R. Oh, M. Park, M. A. K. Rakhimov and J. Ahn, "A Review of Intelligent Self-Driving Vehicle Software Research", KSII TRANSACTIONS ON INTERNET AND INFORMATION SYSTEMS VOL. 13, NO. 11, Nov. 2019.

[9] D. E. Hernandez, S. Blumenthal, E. Prassler, S. Bo and Z. Haojie, "Vision-based road boundary tracking system for unstructured roads," 2017 IEEE International Conference on Unmanned Systems (ICUS), Beijing, 2017, pp. 66-71, doi: 10.1109/ICUS.2017.8278319.

[10] P. Y. Shinzato, V. Grassi, F. S. Osorio and D. F. Wolf, "Fast visual road recognition and horizon detection using multiple artificial neural networks," 2012 IEEE Intelligent Vehicles Symposium, Alcala de Henares, 2012, pp. 1090-1095, doi: 10.1109/IVS.2012.6232175.

[11] X. Liu, G. Wang, J. Liao, B. Li, Q. He and M. Q. -. Meng, "Detection of geometric shape for traffic lane and mark," 2012 IEEE International Conference on Information and Automation, Shenyang, 2012, pp. 395-399, doi: 10.1109/ICInfA.2012.6246837.

[12] Y. Fan, W. Zhang, X. Li, L. Zhang and Z. Cheng, "A robust lane boundaries detection algorithm based on gradient distribution features," 2011 Eighth International Conference on Fuzzy Systems and Knowledge Discovery (FSKD), Shanghai, 2011, pp. 1714-1718, doi: 10.1109/FSKD.2011.6019919.

[13] T. T. Duong, C. C. Pham, T. H. Tran, T. P. Nguyen and J. W. Jeon, "Near real-time ego-lane detection in highway and urban streets," 2016 IEEE International Conference on Consumer Electronics-Asia (ICCE-Asia), Seoul, 2016, pp. 1-4, doi: 10.1109/ICCE-Asia.2016.7804748.

[14] Akhil Suri, “Building an Autonomous Vehicle Part 2: Using Computer Vision to detect lane lines on road”, 8 April 2018, https://medium.com/@akhilsuri194/building-an-autonomous-vehicle-part-2-using-computer-vision-to-detect-lane-lines-on-road-31ea3cda0cbd.
[15] “OpenCV 4.2 documentation”, 20 December 2020, https://docs.opencv.org/4.2.0/.
[16] “Organización Mundial de la Salud, 10 datos sobre la seguridad vial en el mundo”, Julio 2017, https://www.who.int/features/factfiles/roadsafety/es/.
[17] “Día Mundial de las Víctimas de Accidentes de Tráfico”, 18 Noviembre 2018, https://www.rtve.es/noticias/20181118/cada-25-segundos-muere-persona-mundo-accidente-trafico/1840300.shtml.
[18] “NumPy v1.18 Manual”, 24 May 2020, https://numpy.org/doc/1.18/.
[19] “Scikit-video video processing in Python”, 18 September 2018, http://www.scikit-video.org/stable/index.html
[20] “Python”, 24 February 2020, https://www.python.org/downloads/release/python-382/
[21] “Matplotlib v. 3.2.1”, 8 April 2020, https://matplotlib.org/3.2.1/contents.html.
[22] Yuchi Tian, Kexin Pei, Suman Jana, and Baishakhi Ray. 2018. “DeepTest: automated testing of deep-neural-network-driven autonomous cars.”, 40th International Conference on Software Engineering (ICSE ’18). Association for Computing Machinery, New York, NY, USA, 303–314. DOI: https://doi.org/10.1145/3180155.3180220.
[23] B. T. Nugraha, S. Su and Fahmizal, "Towards self-driving car using convolutional neural network and road lane detector," 2017 2nd International Conference on Automation, Cognitive Science, Optics, Micro Electro--Mechanical System, and Information Technology (ICACOMIT), Jakarta, 2017, pp. 65-69, doi: 10.1109/ICACOMIT.2017.8253388.
[24] Wenshuo Gao, Xiaoguang Zhang, Lei Yang and Huizhong Liu, "An improved Sobel edge detection," 2010 3rd International Conference on Computer Science and Information Technology, Chengdu, 2010, pp. 67-71, doi: 10.1109/ICCSIT.2010.5563693.
[25] Xiangyang Xu, Shengzhou Xu, Lianghai Jin and Enmin Song, “Characteristic analysis of Otsu threshold and its applications”, Pattern Recognition Letters, Volume 32, Issue 7, 2011, Pages 956-961, ISSN 0167-8655, https://doi.org/10.1016/j.patrec.2011.01.021.
[26] Z. Chen and X. Huang, "End-to-end learning for lane keeping of self-driving cars," 2017 IEEE Intelligent Vehicles Symposium (IV), Los Angeles, CA, 2017, pp. 1856-1860, doi: 10.1109/IVS.2017.7995975.
[27] S. Chen, S. Zhang, J. Shang, B. Chen and N. Zheng, "Brain-Inspired Cognitive Model With Attention for Self-Driving Cars," in IEEE Transactions on Cognitive and Developmental Systems, vol. 11, no. 1, pp. 13-25, March 2019, doi: 10.1109/TCDS.2017.2717451.
[28] Raja Muthalagu, Anudeepsekhar Bolimera and V. Kalaichelvi, “Lane detection technique based on perspective transformation and histogram analysis for self-driving cars”, Computers & Electrical Engineering, Volume 85 July 2020, 106653, ISSN 0045-7906, https://doi.org/10.1016/j.compeleceng.2020.106653.

Glosario

SWT	Sliding Window Technique
RGB	Red Green Blue
HSL	Hue Saturation Lightness
HSV	Hue Saturation Value
TFG	Trabajo de Fin de Grado